Understanding Privacy Threats in Machine Learning as a Service (MLaaS)

Master Thesis

Gardikis Orestis

Aalborg University
Electronics and IT

Electronics and IT
Aalborg University
http://www.aau.dk

AALBORG UNIVERSITY
STUDENT REPORT

Title:
Understanding Privacy Threats in Machine Learning as a Service (MLaaS)

Project Period:
Fall Semester 2025

Participant(s):
Gardikis Orestis

Supervisor(s):
Qiongxiu Li

Abstract:

Machine Learning as a Service (MLaaS) introduces privacy risks through membership inference attacks, where an adversary determines whether specific data records were used during training. This thesis investigates how differential privacy and regularisation affect model generalisation and privacy leakage across both natural image datasets (CIFAR-10, CIFAR-100) and medical datasets (OCTMNIST, RetinaMNIST, PathMNIST). Three black-box attack paradigms are evaluated: score-based, shadow-model, and label-only transfer attacks. Models are trained under four regimes: non-private, regularised, differentially private (DP-SGD), and DP-SGD combined with regularisation. Overfitting is shown to amplify privacy leakage, while differential privacy and regularisation mitigate it by promoting stability and generalisation. Transfer learning improves robustness in complex datasets, and in smaller medical datasets, DP noise can even enhance performance. The findings demonstrate that privacy and utility are jointly shaped by dataset complexity, model design, and training configuration, emphasising the importance of balanced strategies for secure and reliable MLaaS deployment.

Contents

1 Introduction **1**
1.1 Introduction . . . 1
1.1.1 Privacy as a Central Concern in MLaaS . . . 1
1.1.2 Membership Inference Attacks as a Canonical Threat . . . 2
1.1.3 Regulatory and Ethical Implications . . . 2
1.1.4 Problem Statement, Objectives, and Contributions . . . 3
1.1.5 Thesis Structure . . . 4

2 Machine Learning Background **5**
2.1 Key Machine Learning Definitions . . . 5
2.2 Types of Machine Learning . . . 6
2.3 Model Training and Evaluation . . . 8
2.3.1 Parametric models . . . 9
2.3.2 Overfitting and Underfitting . . . 12
2.3.3 Dataset Splits . . . 13
2.3.4 Non-parametric Models . . . 13
2.3.5 Model Evaluation . . . 14
2.3.6 Optimasation of Parametric Models . . . 15
2.4 Deep Learning . . . 18
2.4.1 Artificial Neural Network . . . 18
2.4.2 Activation functions . . . 21
2.4.3 Convolutional Neural Networks . . . 22
2.5 Privacy in Machine Learning . . . 26
2.5.1 Privacy Threats in Machine Learning . . . 27
2.5.2 Defenses in Machine Learning . . . 29
2.5.3 Evaluating Privacy Leakage in Machine Learning . . . 31

3 Related Work on Membership Inference Attacks **33**
3.1 The Emergence of Membership Inference . . . 33
3.2 Lowering the Barrier: Metric-Based and Loss-Based Simplifications . . . 34
3.2.1 Salem et al. (2018): Metric-Based Attacks and the ML-Leaks Paradigm 34

3.2.2 Yeom et al. (2018): Loss-Based Thresholding . . . 35
3.3 White-Box and Gradient-Based Attacks . . . 35
3.3.1 Nasr et al. (2019): Privacy Analysis Through White-Box Inference . . . 35
3.3.2 Leino & Fredrikson (2020): Calibration and Memorization . . . 36
3.4 Label-Only, Contrastive, and Memorization-Aware Attacks . . . 36
3.4.1 Liu et al. (2021): Contrastive Representation Leakage . . . 36
3.4.2 Label-Only Attacks . . . 37
3.4.3 Carlini et al. (2022): Membership Inference from First Principles . . . 38
3.4.4 Li et al. (2024): Reassessing Privacy Through Memorization . . . 38
3.5 Summary and Conceptual Taxonomy . . . 39

4 Methodology **41**
4.1 Experimental scope and principles . . . 41
4.1.1 Training regimes . . . 41
4.1.2 Evaluation metrics . . . 42
4.2 Score-based membership inference attack . . . 42
4.2.1 Threat model . . . 42
4.2.2 Attack formulation . . . 42
4.2.3 Datasets and preprocessing . . . 43
4.2.4 Architectures and training regimes . . . 43
4.2.5 Evaluation . . . 43
4.3 One-shadow model membership inference attack (CIFAR-10) . . . 44
4.3.1 Threat model . . . 44
4.3.2 Dataset split . . . 44
4.3.3 Model architecture . . . 44
4.3.4 Attack construction . . . 45
4.3.5 Training regimes and evaluation . . . 45
4.4 Transfer (label-only) membership inference attack (CIFAR-10) . . . 45
4.4.1 Threat model . . . 45
4.4.2 Dataset split . . . 46
4.4.3 Model architecture . . . 46
4.4.4 Attack procedure . . . 47
4.4.5 Training regimes and evaluation . . . 47

5 Results **49**
5.1 Score-Based Attack . . . 50
5.1.1 CIFAR-10 . . . 50
5.1.2 CIFAR-100 . . . 52
5.1.3 OCTMNIST . . . 55
5.1.4 RetinaMNIST . . . 57
5.1.5 PathMNIST . . . 59
5.2 One Shadow-Model Attack . . . 61

5.2.1 CIFAR-10 . . . 62
5.2.2 PathMNIST . . . 63
5.3 Transfer Attack . . . 65
5.3.1 CIFAR-10 . . . 65
5.3.2 PathMNIST . . . 67

6 Discussion **69**
6.1 Cross-Dataset Patterns in Membership Leakage . . . 69
6.2 Utility–Privacy Trade-off and Practical Deployability . . . 71
6.3 Evaluating ROC vs TPR@FPR Metrics . . . 72
6.4 Attack Surface Under Different Adversary Capabilities . . . 73
6.5 Influence of Model Capacity, Pretraining, and Data Regime . . . 74
6.6 Limitations and Threat Model Gaps . . . 75
6.7 Implications for MLaaS and Future Work . . . 76

7 Conclusion **78**
7.1 Answers to the Research Questions . . . 79
7.2 Contributions and Significance . . . 80
7.3 Closing Reflection . . . 81

Bibliography **82**

A Supplementary figures and notebooks **86**
A.1 CIFAR-10 . . . 86
A.1.1 Baseline (no dropout, no early stopping) . . . 86
A.1.2 Regularised (dropout + early stopping) . . . 88
A.1.3 DP-SGD (TanhCNN) . . . 90
A.1.4 DP-SGD + Regularised (LeakyDropCNN + ES) . . . 96
A.2 Cifar-100 . . . 104
A.2.1 ResNet18 /w Transfer . . . 104
A.2.2 Resnet . . . 124
A.2.3 WideResNet . . . 132
A.3 OCTMNIST . . . 140
A.3.1 Baseline . . . 140
A.3.2 Regularised . . . 142
A.3.3 DP no Regularization . . . 144
A.3.4 DP + Regularisation . . . 150
A.4 RetinaMNIST . . . 156
A.4.1 Baseline . . . 156
A.4.2 Regularized . . . 159
A.4.3 DP no Regularisation . . . 160
A.4.4 DP + Regularisation . . . 168

A.5 PathMNIST 176
A.5.1 Baseline 176
A.5.2 Regularised 178
A.5.3 DP no Regularisation 180
A.5.4 DP + Regularisastion 189

B Shadow + Transfer Notebooks **197**
B.1 Shadow Experiments 197
B.1.1 CIFAR-10 197
B.1.2 PathMNIST 198
B.2 Transfer Experiments 198
B.2.1 CIFAR-10 198
B.2.2 PathMNIST 199

Chapter 1

Introduction

1.1 Introduction

Machine learning (ML) has rapidly evolved from a research discipline into a foundational technology underpinning modern artificial intelligence (AI). By learning directly from data, ML systems uncover complex patterns and improve predictive accuracy without explicit programming [1]. This data-driven paradigm has enabled advances across domains such as natural language processing, computer vision, healthcare, and cybersecurity. However, as ML systems increasingly operate on sensitive information, concerns about privacy, transparency, and accountability have become central to their responsible deployment.

The emergence of *Machine Learning as a Service* (MLaaS) has accelerated this trend. Cloud-based platforms such as Google Cloud AI, Amazon SageMaker, and Microsoft Azure ML democratize access to large-scale ML infrastructure by allowing users to train and deploy models without managing the underlying resources. While MLaaS greatly simplifies development, it also introduces new risks: models trained on personal or proprietary data are exposed to external queries through public APIs, creating potential vectors for privacy leakage. Understanding and mitigating these risks is therefore critical to the secure adoption of MLaaS in sensitive sectors.

1.1.1 Privacy as a Central Concern in MLaaS

The utility of ML systems depends on the quality and volume of their training data, which frequently includes personally identifiable or confidential information. Even when explicit identifiers are removed, latent patterns in the data may still encode sensitive attributes. Providers often assume that once trained, models abstract away individual records, revealing only aggregate behavior. However, research has shown that this assumption does not hold universally: ML models can *memorize* training samples and inadvertently disclose them through their outputs [2, 3]. This challenges the conventional view of models as neu-

tral statistical abstractions and reframes them as potential carriers of private information.

The risk is amplified in MLaaS settings, where adversaries typically interact with deployed models in a *black-box* manner. By issuing repeated queries and analysing confidence outputs, an attacker may infer properties of the training data or determine whether specific samples were included in it. Such disclosures not only threaten user trust but, in regulated domains like healthcare or finance, may constitute direct violations of data protection legislation.

1.1.2 Membership Inference Attacks as a Canonical Threat

Among the various privacy risks in ML, *Membership Inference Attacks* (MIAs) represent the most fundamental test of confidentiality. In an MIA, the adversary seeks to determine whether a given record was part of a model's training dataset. Although this binary question appears simple, its implications are significant: confirming a patient's inclusion in a medical training dataset can reveal a diagnosis, identifying a financial record can disclose a transaction history. As Hu et al. observe, membership inference undermines privacy at its core, since it exposes the very presence of individuals in a dataset [3].

The evolution of MIAs reflects their increasing practicality. Shokri et al. [2] first formalised the attack using shadow models to approximate target behaviour through confidence scores. Subsequent studies demonstrated that even weaker adversaries—those without detailed model knowledge—can perform effective MIAs using simple confidence thresholds [4]. More recent work established that MIAs remain viable even when only predicted labels are available [5, 6], underscoring their relevance to real-world MLaaS systems. Parallel research by Carlini et al. [7] re-examined the foundations of privacy evaluation, proposing likelihood-based attacks such as LiRA that quantify leakage more precisely in low false-positive regimes.

1.1.3 Regulatory and Ethical Implications

The privacy vulnerabilities exposed by MIAs intersect directly with data protection law and ethics. Under the General Data Protection Regulation (GDPR), any information relating to an identifiable individual is classified as personal data. A trained model that reveals the membership status of individuals may therefore itself constitute personal data [8]. This interpretation places models within the same regulatory framework as the datasets used to train them, introducing obligations for transparency, accountability, and lawful processing. Beyond legal definitions, the ethical dimension is clear: individuals contribute data under the expectation that their participation will not expose them to harm. If models leak information about specific contributors, this expectation and the societal trust in AI systems is undermined.

1.1.4 Problem Statement, Objectives, and Contributions

Machine learning models deployed through Machine Learning as a Service (MLaaS) platforms can unintentionally expose sensitive information about their training data. A prominent threat is the *membership inference attack* (MIA), where an adversary attempts to determine whether a specific record was part of the training set. Such attacks exploit differences in confidence, loss, or prediction behaviour between training and unseen examples, potentially violating data confidentiality even when the underlying data are not directly accessible.

This thesis investigates how these privacy leakages occur and how they can be mitigated without excessively compromising model utility. The focus lies on the combined influence of regularisation and differential privacy (DP) on model generalisation and resistance to membership inference across both natural and medical image domains. The study examines three adversarial settings that represent progressively weaker forms of access: a **score-based** attack relying on confidence scores, a **shadow-model** attack trained on auxiliary data, and a **transfer-based** (label-only) attack operating on discrete outputs. Together, these attacks approximate realistic MLaaS conditions, where the degree of output visibility directly shapes privacy risk.

The research is guided by the following objectives:

1. To empirically evaluate membership inference attacks across natural (CIFAR-10, CIFAR-100) and medical image datasets (OCTMNIST, RetinaMNIST, PathMNIST) under different adversarial assumptions.

2. To analyse the relationship between overfitting, regularisation, and privacy leakage, using non-member accuracy as the principal measure of model utility.

3. To assess the impact of differentially private training (DP-SGD) on membership inference resistance, quantifying both privacy budgets (ε) and classification performance.

4. To examine how architectural choices, including model capacity and transfer learning, interact with privacy noise and influence the privacy–utility frontier.

In line with these objectives, the thesis addresses three central research questions:

RQ1: How do different privacy attacks perform against machine learning models trained on medical and natural image datasets with varying levels of sensitivity and structure?

RQ2: What is the extent of privacy leakage due to membership inference in such models, and which metric is more appropriate for quantifying this leakage?

RQ3: Which mitigation strategies can effectively reduce privacy leakage while maintaining acceptable model utility?

This thesis contributes to the empirical and practical understanding of privacy in machine learning through the following achievements:

- **Cross-domain evaluation of attack models.** The study provides a unified, multi-dataset comparison of three membership inference attack types applied to models trained under four distinct regimes: standard, regularised, DP-SGD, and DP-SGD with regularisation. This design enables a systematic comparison of adversarial strength and defensive effectiveness across domains.
- **Insight into privacy–utility dynamics.** The experiments show that privacy leakage closely tracks overfitting and that both regularisation and differential privacy reduce leakage by improving generalisation. Non-member accuracy is introduced as a robust indicator of model utility, allowing consistent comparison of privacy trade-offs across datasets.
- **Architectural and domain-level findings.** Transfer learning stabilises DP-SGD on complex datasets such as CIFAR-100, while smaller and more homogeneous medical datasets exhibit lower leakage even without privacy noise, revealing domain-dependent privacy resilience.
- **Clarification of evaluation metrics.** The study distinguishes between AUC and TPR@FPR as complementary privacy metrics, showing that high theoretical separability does not necessarily imply actionable leakage when low-FPR detection remains near zero.
- **Integration of regulatory perspective.** The results are framed within the concept of Privacy by Design, linking algorithmic stability and differential privacy to practical compliance and trustworthiness in MLaaS systems.

1.1.5 Thesis Structure

The remainder of this thesis is organised as follows. Chapter 2 introduces foundational concepts in machine learning, privacy attacks, and defence mechanisms. Chapter 3 surveys membership inference attacks, tracing their methodological evolution from shadow models to label-only settings. Chapter 4 outlines the experimental design, including datasets, architectures, training regimes, and privacy accounting. Chapter 5 presents the empirical results, while Chapter 6 interprets these findings across datasets and discusses their regulatory and ethical implications. Finally, Chapter 7 concludes the thesis by summarising key insights and identifying future research directions.

The Appendix compiles all supplementary material referenced throughout the thesis, including full training and validation curves, ROC plots, confidence distributions, and comparative figures for each configuration. It also provides direct links to the corresponding Google Colab notebooks and scripts, ensuring full transparency and reproducibility of the experimental results.

Chapter 2

Machine Learning Background

Machine learning (ML) is the study of data–driven models that learn relationships between inputs and outputs from sampled data, rather than being specified by hand. Early ideas of machines that can learn trace back at least to Ross (1937) [9], and were popularized by Samuel's checkers program in 1959 [10]. Since then, advances in algorithms, data availability, and computation have progressively reduced prior constraints, enabling solutions to increasingly complex tasks.

Today, ML underpins a broad spectrum of applications and infrastructures: perception and language (computer vision, speech recognition, natural language processing), decision and control (recommendation, forecasting, operations research, robotics), cyber–physical and industrial systems (predictive maintenance, quality control, autonomous systems), security and trust (anomaly detection, malware classification), scientific discovery (protein folding, materials design), and large–scale cloud offerings under the umbrella of Machine Learning as a Service (MLaaS). In each case, the core objective is the same: learn a mapping that generalizes from observed data to unseen instances with reliable performance and calibrated uncertainty.

This chapter introduces the mathematical and theoretical foundations needed for the remainder of the thesis. We formalize learning problems and notation, review optimization and generalization concepts, and discuss evaluation metrics and deployment interfaces that are later shown to be relevant for privacy. These fundamentals provide the basis for analyzing how training dynamics and model behavior can expose sensitive information in downstream settings.

2.1 Key Machine Learning Definitions

Machine learning (ML) is a meticulous domain that uses an excessive amount of terminology derived from mathematics, statistics and programming. To make the next chapters less facile, in this section we introduce some key definitions. Frequently, a set of input data, $\boldsymbol{X}$ is collected along with their outcomes, $\boldsymbol{y}$ and the goal is to create a model that

learns from the data which after the learning process is able to make a prediction of the outcome $\hat{\boldsymbol{y}}$ given some unseen input data. The data can be of any form including numerical values, images, audio, video or a combination of the before-mentioned. A convenient way to represent the data is by constructing a design matrix $\boldsymbol{X}$ where each row represents a single example of the data, e.g. a patient, where each column represents the characteristics of the example e.g. patient's name, age or weight. The number of rows in $\boldsymbol{X}$ is then the collected sample size $\boldsymbol{X}$ while the number of columns represents the total number of features or characteristics, for each sample, $\boldsymbol{D}$. Similarly, the collected outcomes from each input example can be expressed as an $\boldsymbol{N}$-dimensional vector, where each element represents the outcome of the nth example, if $\boldsymbol{n}$ = 1,...,$\boldsymbol{N}$. In a mathematical sense, the input data and their outcome can be written as:

$$\mathbf{X} = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1D} \\ x_{21} & x_{22} & \cdots & x_{2D} \\ \vdots & \vdots & \ddots & \vdots \\ x_{N1} & x_{N2} & \cdots & x_{ND} \end{bmatrix}, \qquad \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix}$$

where each row of $\boldsymbol{X}$ can be also be written in the form of a D-dimensional vector:

$$x_n = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_D \end{pmatrix}^T$$

In ML, we conceptualise an *algorithm* as a mathematical technique derived by statisticians and mathematicians for a particular task implemented in code which will help us find a relationship between the inputs and the output. Different algorithms make different assumptions about the nature of this relationship and how it can be learned. For example, $y = w_0 + xw_1$ is the linear regression equation for a single feature, which can be easily written in code, making it an ML algorithm. For this example, an ML algorithm assumes a linear relationship between x and y, which can be estimated through the parameters w_0 and w_1. Hence, the above algorithm would estimate the weights w_0, w_1 that best map the input data to the output variables y. In order to do so, we would need to have some pairs of input, output examples x, y available, and then we could solve the above equation with respect to w_0 and w_1 to find their estimations. We can define a model as an equation which is formed by finding out the parameters w_0, w_1 in the equation of the algorithm, i.e. for this example y= $\hat{w}_0$ + $x\hat{w}_1$. More, generally, a model is created by using available data and an algorithm which,will enable us to find a way to predict new input data.

2.2 Types of Machine Learning

Machine Learning can solve numerous types of tasks, and subsequently, there are several variations of algorithms available. One convenient approach to categorise the vast number

of algorithms is across four categories:

1. **Supervised Learning (SL):** In SL, models are trained using labelled examples, i.e. a design matrix of inputs X where the desired outputs y are known. There are two types of SL tasks which are determined by the nature of the outputs.

   (a) **Regression:** In a regression problem, the outcome is continuous, or in other words, the purpose is to predict a numeric value based on a set of inputs. An example of regression is to predict the price of a house based on some of its features, e.g. number of bedrooms, location, year of construction.

   (b) **Classification:** In classification the outcome is discrete, i.e. its value is a class or a label describing a particular example, e.g. whether an example is a dog, a cat or a bird. In other words, the goal in classification is to predict a discrete class output based on a set of inputs.

   It is possible to further split classification based on the number of classes:

   - **Binary classification:** Outcome can only take one of two possible values. Example: automated distinction of healthy patients and patients with cancer.
   - **Multi-class classification:** More than two classes. Example: label assignment of different flower species based on a set of features.

   There is a wide range of SL algorithms that have been developed over the years, thus one should choose an algorithm based on the nature of the data. Some of the most critical SL algorithms are linear regression, logistic regression, k-nearest neighbors (KNN), decision trees and random forests, support vector machines (SVMs), naive Bayes, neural networks [11].

2. **Unsupervised Learning (UL):** In UL, models are trained using unlabeled examples, i.e. inputs X where the desired output vector y is unknown, or we do not want to use it. Instead of using labels, the model is allowed to work on its own to discover information. Different task categories fall under UL.

   (a) **Clustering:** In clustering, the goal is to find structures and patterns in a collection of uncategorised data, i.e., those algorithms find natural groups/clusters in the data. Some important clustering algorithms are K-Means and hierarchical cluster analysis [11]

   (b) **Anomaly Detection:** Such tasks attempt to identify outliers in a dataset, i.e., observations that differ from the dataset's normal behaviour [12]

   (c) **Dimensionality Reduction:** The idea of dimensionality reduction is about representing data in a lower-dimensional space where certain properties are preserved as much as possible. It can be used to visualise high-dimensional data in a lower-dimensional space.

One way to achieve the above is by *feature transformation*, i.e., constructing new features based on a group of features, also known as feature extraction. Some important algorithms include Principal Component Analysis (PCA), Kernel PCA, Metric Multidimensional Scaling (MDS), and Isomap.[13, 14]

3. **Semi-supervised Learning (Semi-SL):** In Semi-SL, models are trained using partially labelled examples, i.e., inputs X where the desired output vector y is known only for some instances. Most Semi-SL algorithms are a combination of both supervised and unsupervised approaches.

   This method is especially useful when extracting similar features from the data is difficult or costly, and labelling examples is a time-consuming process for experts. An example is in medical imaging such as CT scans, where labelling patients as healthy or cancerous by a radiologist can be time-intensive. By labelling only a small portion of the patients, a neural network can still gain significant accuracy improvements compared to being completely unsupervised.

4. **Reinforcement Learning (RL):** Such algorithms have a distinctive training process. The learning system (agent) observes the environment and selects and performs actions based on which it receives rewards or penalties. The agent must learn by itself to choose the best strategy (policy) in order to maximise the reward over time. A policy defines what action the agent should choose when it is in a given situation.[15]

2.3 Model Training and Evaluation

We now focus on the technical aspects of supervised learning (SL): (i) identifying the task type, (ii) selecting an algorithm and training a model, and (iii) measuring performance.

For example, we can train a model on the CIFAR-10 dataset, where the goal is to categorise images into ten groups based on their content (e.g., airplane, automobile, bird, cat, deer, dog, frog, horse, ship, and truck). In this case, the type of the ML task will be multi-class classification.

To make the concepts that follow more comprehensible we will focus on a single input example,

$$x_n = (x_1, x_2, \ldots, x_D),$$

which represents the nth row from an $N \times D$ design matrix, X, and the corresponding output y_n. To reduce the mathematical notation, in the following paragraphs we will refer to x_n and y_n as x and y respectively, unless specified otherwise.

We start with the assumption that x and y have an unknown joint probability density function $p(x, y)$, hence for some fixed values of x, y follows an unknown conditional distribution $p(y|x)$. In a probabilistic sense, the goal is to estimate the conditional distribution of y given the input data.

Usually, we are only interested in a point estimation of y rather than its distribution. Hence, we are looking for a statistical function of x that can approximate the value of y, based on the data it has seen. In other words, we are looking for a prediction function $f(x)$ that provides an estimate of y, $\hat{y}$, for any possible value of x, i.e.,

$$y \approx f(x) = \hat{y}.$$

More generally, ML is learning a function f that maps x to y, and in order to achieve that, an algorithm learns this mapping function from the training data. However, as f is unknown, one has to evaluate different algorithms and make assumptions in order to approximate the underlying f. The assumptions we make streamline the learning process; however, they can also limit it. SL algorithms segregate into two categories based on the assumptions made, which are called *parametric* and *non-parametric*.

2.3.1 Parametric models

A learning model that summarises data with a set of parameters of fixed size (independent of the number of training examples) is called a parametric model. No matter how much data you throw at a parametric model, it will not change its mind about how many parameters it needs. [16].

In the parametric framework, we need to specify the form of f according to the relationship between x and y, which can be either linear or non-linear. For example, if we assume that f is linear in x, then:

$$f(x) = w_0 + w_1 x_1 + w_2 x_2 + \cdots + w_D x_D.$$

An example of a non-linear relationship, i.e., where f is not linear in x, could be the sigmoid function:

$$f(x) = \sigma(x) = \frac{1}{1 + e^{-(w_0 + w_1 x_1 + w_2 x_2 + \cdots + w_D x_D)}}$$

where $\mathbf{w} = (w_0, w_1, \ldots, w_D)$ is a vector containing the parameters of the model, also known as the *weight vector*.

In the linear formula of f, we can observe that each element from the weight vector, except one, is multiplied with a feature of the input vector. The term w_0 is called the *bias term*, and in a single dimension (i.e., if x has only one feature), it can be interpreted as the value of the intercept, that is, the value of $f(x)$ when $x = 0$. For D-dimensional inputs x, it can be thought of as the height of the function's $f(x)$ plane.

The second step in parametric modelling is to learn the parameters of the function from the training data. To achieve that, we first need to understand the upcoming ideas. In a loose sense, each prediction $\hat{y}$ made by $f(x)$ can be really close to, or far from, the corresponding real value of y. In any case, we need a measure that can help us evaluate how close or far our prediction made by $f(x)$ actually is. These metrics are called *loss functions* $L(y, \hat{y})$, and are written in terms of the true value y and the prediction $\hat{y}$.

There is a rich collection of loss functions to choose from. For regression tasks, the most common is the squared error loss, which can be expressed as:

$$L(y, \hat{y}) = (y - \hat{y})^2$$

As mentioned, a loss function corresponds to a single prediction. However, in ML we usually have a collection of more than one training instance. Given an $N \times D$ design matrix X, we obtain N input–output pairs $(x_n, y_n)\}_{n=1}^{N}$ drawn independently from $p(x, y)$, and we fit a prediction function $f(X)$ on the observed data which results into a prediction vector:

$$\hat{y} = \begin{pmatrix} \hat{y}_1 \\ \hat{y}_2 \\ \vdots \\ \hat{y}_N \end{pmatrix}, \text{with N corresponding loss functions } \{L(y_n, \hat{y}_n)\}_{n=1}^{N}$$

A function which can evaluate the quality of predictions generated by the model's prediction function is called the *cost function*, $C(\mathbf{w})$. The cost function is the expectation of the loss function with respect to $p(x, y)$. That is:

$$C(\mathbf{w}) = \mathbb{E}_{y,\hat{y}}[L(y, \hat{y})] = \mathbb{E}_{p(x,y)}\big[L\big(y, f(x)\big)\big] = \int L\big(y, f(x)\big)\, p(x, y)\, dx\, dy.$$

The key point of the cost function is that the loss is a function of a given x_n and the weight vector $\mathbf{w}$. Parametric ML focuses on the optimisation of the model's parameters. That is, to estimate the weight vector which minimises the cost function:

$$\hat{\mathbf{w}} = \arg\min C(\mathbf{w}).$$

In real-life problems, the cost function lies in a multidimensional space, making the computation of the cost's integral infeasible or extremely hard. However, it can be approximated by an average over the training loss instances, which is referred to as the *training cost*. That is:

$$C(\mathbf{w}) = \int L\big(y, f(x)\big)\, p(x, y)\, dx\, dy \;\approx\; \frac{1}{N} \sum_{n=1}^{N} L(y_n, \hat{y}_n) = C_{\text{train}}(\mathbf{w}).$$

In summary, the parametric training process of a model in ML, assuming N input–output pairs $\{(x_n, y_n)\}_{n=1}^{N}$, is to find a prediction function $f(x_n)$, define a loss function, and estimate the weight vector which minimises the cost function.

For regression, some well-known training cost functions are:

- **Mean Squared Error (MSE):**

$$\text{MSE} = \frac{1}{N} \sum_{n=1}^{N} \big(y_n - f(x_n)\big)^2$$

- **Mean Absolute Error (MAE):**

$$\text{MAE} = \frac{1}{N}\sum_{n=1}^{N}|y_n - f(x_n)|$$

In classification, while most of the terminology and ideas behind regression still apply, there are some significant differences. In contrast to regression models, classification algorithms (classifiers) are probabilistic models in the sense that instead of trying to approximate the output values directly, they estimate the probability that a training instance x_n belongs to a class c.

The way the output of a classifier becomes a probability is by bounding the outcome value between 0 and 1. For example, a well-known binary classifier, *logistic regression*, uses a non-linear prediction function:

$$f(x) = \sigma(\mathbf{w}^\top x) = \frac{1}{1+e^{-\mathbf{w}^\top x}} \ \in [0,1]$$

Then, the model's prediction interpretation is equivalent to

$$P(y = c \mid x, \mathbf{w}) \ \approx \ f(x).$$

For such tasks, it is common to use the *maximum likelihood* as the training cost function. In that way, the optimisation task is to maximise the probability of the data given $\mathbf{w}$:

$$P(y \mid x, \mathbf{w}) = \prod_{n=1}^{N} P(y_n \mid x_n, \mathbf{w}).$$

Equivalently, instead of using the maximum likelihood with respect to $\mathbf{w}$, it is more practical to minimise the *negative log-likelihood*, also known as *cross-entropy*. For binary classification, the cross-entropy is given by:

$$-\sum_{n=1}^{N}\Big\{y_n \log(p_n) + (1-y_n)\log(1-p_n)\Big\}$$

where $y_n \in \{0,1\}$ and $p_n = f(x_n, \mathbf{w})$, which in the case of logistic regression is equal to $\sigma(\mathbf{w}^\top x_n)$.

For multi-classification tasks, we calculate a separate cost for each class label per observation and sum the result. Then the cross-entropy is given by:

$$-\sum_{n=1}^{N}\sum_{c=1}^{C} y_{n,c}\log(p_{n,c})$$

where $C > 2$ is the number of classes, $y_{n,c}$ is a binary indicator (0 or 1) denoting whether the class label c is the correct classification for observation n, and $p_{n,c} = P(y_n = c \mid x_n, \mathbf{w})$ is the predicted probability that observation n belongs to class c.

2.3.2 Overfitting and Underfitting

Several complications may occur during training. If the underlying relationship of the data is too complicated or our training data are too noisy, the estimated weights of our model may adjust to the noise in such an extent that the model loses its ability to predict or generalise well to new, unseen data. An opposite scenario is that the model is too simple, it does not capture the underlying structure of the data, and thus it cannot predict well during training nor generalise to new data. The first phenomenon is known as *overfitting*, while the latter is known as *underfitting*[15]. There are plenty of solutions to tackle overfitting. One is to simplify the model by selecting fewer input features. Other options are to gather more training data, reduce the noise by removing possible outliers, or to constrain the model.

Regarding the latter, it is possible to apply different regularisation approaches in the cost function to penalise the weights for taking extreme values. In other words, one can adjust the trade-off between generalisation and extreme weight fits. One way to do this is *L2 regularisation*, where a hyperparameter λ is applied to the weights in order to penalise extreme fits during the cost function minimisation. For example, if the MSE training cost function is:

$$C_{\text{train}}(\mathbf{w}) = \text{MSE} = \frac{1}{N}\sum_{n=1}^{N}\big(y_n - f(x_n)\big)^2$$

Then the L2 regularisation has the form:

$$C_{\text{train},\lambda}(\mathbf{w}) = \frac{1}{N}\sum_{n=1}^{N}\big(y_n - f(x_n)\big)^2 + \lambda\sum_{d=1}^{D} w_d^2$$

where $\lambda \in [0, \infty)$. If $\lambda = 0$, then the two above functions are identical, meaning there is no regularisation, whereas the more λ increases, the smaller the weights will be.

To prevent underfitting, one can select a model with more features, create more insightful features (*feature engineering*), or reduce the value of the regularisation parameter λ.

A summary of what has been discussed so far is that after gathering the data, in parametric supervised learning (SL), we choose a prediction function and estimate the weights by minimising the cost function while simultaneously constraining the extremeness of the weight values. Nevertheless, the most important criterion for a good model is its ability to generalise well to unseen data. Moreover, we may need to test several values of λ in order to compare which one is the best, or perhaps test different algorithms to see which yields better predictions.

The way to achieve all of the above is to split our dataset into three new datasets. Each of these new datasets has a different name and a different purpose, presented in the nect section.

2.3.3 Dataset Splits

1. **Training Set:** It contains most of the input–output pairs, and it is the set where all the training process described occurs.

2. **Validation Set:** After training the parameters of different models or testing different values for the regularisation parameter λ, we evaluate the performance of all the models on some held-out data, called *validation data*. The evaluation is done by calculating a cost function, using the estimated weights found during training, along with the validation data. The cost function used on the validation set is also called the *validation error*, and it may be the same function used during training (e.g., MSE), but a different one such as MAE can be used as well. The model which reports the lowest validation error (i.e., the smallest cost function value) is the one we should choose.

3. **Test Set:** After choosing our model, we need a set where we can report the *generalisation error* of our chosen model. In other words, the test set is only used to indicate how well our already chosen model predicts unseen data.

It is important to note that in case one observes the generalisation test error and then decides to adjust the model in a way that will decrease that error, the test data are no longer unknown and permanently loses its purpose as any change made on the model is overfitting to new data. Occasionally, the sample size may be limited and thus, splitting the available dataset into threesubsets can decrease the training set size in such an extend that it can affect the generalisation ability of the trained model.

2.3.4 Non-parametric Models

Non-parametric models make very few assumptions about the mapping function f, as opposed to theparametric ones. Furthermore, non-parametric methods can also solve both regression and classification tasks, and their performanceimprovement is analogous to the amount of available data and arean ideal picking choice when the is no prior knowledge about the data.

Non-parametric methods seek to best fit the training data in constructing the mapping function, whilst maintaining some ability to generalise to unseen data. As such, they are able to fit a largenumber of functional forms [16].

A common misconception is that non-parametric algorithms do not have any parameters. In aparametric model, we have a finite number of predefined parameters, whereas, in non-parametric models, the number of parameters is potentially infinite, as the complexity increases according to the number of training data. However, the form of the parameters is not determined before training. If a non-parametric model is left completely unconstrained, it will mimic the structure of the data in the extent of overfitting. To prevent such

cases, we should apply a form of regularisation applicable to the non-parametric framework. Each non-parametric algorithm follows a different approach, and thus there is not a general guideline for using them.

2.3.5 Model Evaluation

Usually after training a classifier (parametric or not) on the training set, we evaluate its generalisation ability on the test set by using a confusion matrix. A confusion matrix is a summarised table containing the predictions of the classifier, where the number of correct and wrong predictions for each class is summarised as a count. The confusion matrix gives an intuition regarding the overall performance and the type of errors of the classifier. In a binary classification task where class 1 is denoted as positive, and a class 2 is denoted as negative, the confusion matrix would look like:

	Class 1 predictions	**Class 2 predictions**
Class 1 actual	TP	FN
Class 2 actual	FP	TN

Table 2.1: Confusion matrix for binary classification

where ***TP***: number of true positives, i.e. observations that are positive and were predicted as positive.

FN: number of false negatives, i.e. observations that are positive and were predicted as negative.

TN: number of true negatives, i.e. observations that are negative and were predicted as negative.

FP: number of false positives, i.e. observations that are negative and were predicted as positive.

The table 2.1 is an example of binary classification results, however confusion matrices can extend to multi-classification tasks following the same notion.

Through the content of confusion matrix we can further calculate the following metrics:

- **Accuracy:** It is defined as the total number of correct predictions made by the model, $(TP + TN)$, divided by the total number of predictions, $(TP + FN + FP + TN)$:

$$\text{Accuracy} = \frac{TP + TN}{TP + FN + FP + TN}.$$

  In some cases, accuracy can be a misleading metric. For instance, when dealing with extremely imbalanced data (e.g., 99% of data is class 1 and 1% of data is class 2), the accuracy is expected to be high. Even if one were to randomly predict a class, chances are that 99% of the time, class 1 would be the right one.

- **Recall:** The total number of true positives (TP) divided by the total number of actual positives $(TP + FN)$. This metric evaluates the ability of a model to find all the relevant cases within a dataset for each class:

$$\text{Recall} = \frac{TP}{TP + FN}.$$

- **Precision:** The total number of true positives (TP) divided by the total number of predicted positives $(TP + FP)$. This metric evaluates the ability of a model to identify only the relevant data points:

$$\text{Precision} = \frac{TP}{TP + FP}.$$

- **F1-Score:** This metric is a blend of precision and recall, also referred to as the harmonic mean of precision and recall:

$$F_1 = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}.$$

2.3.6 Optimasation of Parametric Models

The essence of a parametric model's learning process is the selection of its parameters along with a cost function which we then want to optimise. As discussed in Section 2.3, during training, we seek the weights and the bias that minimise the training cost. It is known from calculus that in order to find the minimum of a function, in this case $C_{\text{train}}(\mathbf{w})$, we can set the gradient vector of partial derivatives with respect to the parameter vector equal to zero and find the global minimum. For a simplified model such as linear regression or logistic regression, it is indeed possible to find the best fit of these weights based on the training set. This is because the cost function has the property of convexity.

Convexity:

We call a cost function convex in the vector of weights if, when taking two points of the function for $\mathbf{w}$ and $\mathbf{w}'$ with $\mathbf{w} < \mathbf{w}'$, the line between $C(\mathbf{w})$ and $C(\mathbf{w}')$ lies above the surface of the function. Formally:

$$C\big(a\mathbf{w} + (1-a)\mathbf{w}'\big) \;\leq\; a\,C(\mathbf{w}) + (1-a)\,C(\mathbf{w}'), \quad \forall\ (\mathbf{w}, \mathbf{w}'),\ 0 \leq a \leq 1.$$

A useful property of a convex cost function is that there exists an optimisation method that can track the parameters minimising the cost function as much as possible. On the contrary, if a cost function is not convex, then the optimisation process may not be able to find the global minimum, but instead only a local minimum, depending on the initialisation.

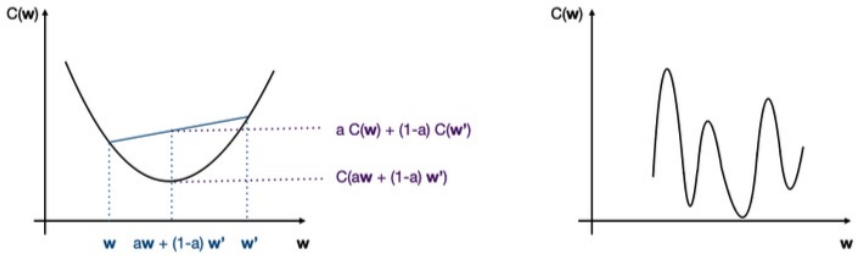


Figure 2.1: A convex function (left) versus a non-convex function (right).

That being said, we can imagine several scenarios where much more complex models are used, thus more parameters, making the calculation of the training cost's gradient extremely hard. To alleviate these situations and still be able to solve many optimisation problems, we use gradient-based methods. The highlight of those methods is the usage of gradient descent, (GD), which is an iterative algorithm with the following structure:

Algorithm 1 Gradient Descent

1: Initialise the weight vector $\mathbf{w}$ to a fixed value
2: **for** $t = 1$ to T **do**
3: $\quad \mathbf{w} \leftarrow \mathbf{w} - \eta \, \nabla_{\mathbf{w}} C_{\text{train}}(\mathbf{w})$

Instead of calculating the weight vector that minimises the training cost, gradient descent starts from an arbitrary point on the cost function. It calculates $-\nabla_{\mathbf{w}} C_{\text{train}}(\mathbf{w})$, i.e., the gradient direction indicating the direction where the cost decreases the most steeply. Once the direction is found, we need to choose the step size we want our weights to move towards that downward direction. That step size is represented by η or α and is called the *learning rate*. After calculating the gradient direction multiplied by the learning rate, the algorithm updates the parameter vector accordingly, with the gradient re-evaluated before each update. Finally, it repeats this process for T iterations with the goal of convergence to a minimum.

It is important to note that a small learning rate leads to slower convergence, while a large one has the risk of missing the minimum, as shown in Figure 2.2. The learning rate can be constant or adaptive during the optimisation process; for example, start with a large value 3which decreases as the gradient of the cost gets closer to zero.

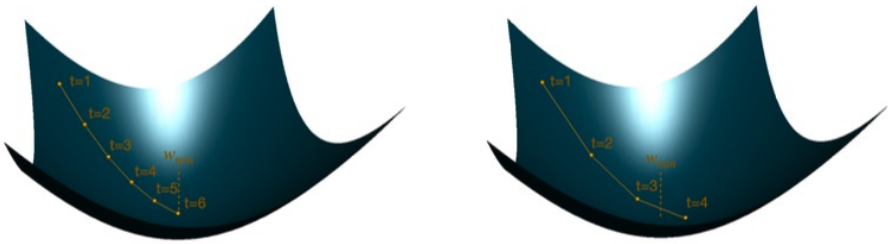


Figure 2.2: small constant learning rate (left) versus large constant learning rate (right).

In traditional gradient descent (GD), each parameter vector update $\nabla_{\mathbf{w}} C_{\text{train}}(\mathbf{w})$ uses the whole training dataset. Such methods are also known as *batch methods*. This is not ideal for ML purposes, since a single parameter update requires the calculation over the entire training set, which usually consists of thousands of training examples.

Typically, the calculated gradient in each update is large, making it easy for the optimiser to miss the minimum. Instead, a more practical way is to use a single training example or a minibatch of examples, and try to improve the model a little, then train on the next minibatch of examples, and so on until the method has used the whole training dataset. Each full training round is called an *epoch*.

These methods are called *stochastic* or *minibatch methods*. Stochastic Gradient Descent (SGD) is a minibatch method which performs an update based on the average gradient. More specifically, the algorithm of SGD is:

Algorithm 2 Stochastic Gradient Descent

1: Initialise $\mathbf{w}$
2: **for** $t = 1$ to T **do**
3: **for** $b = 1$ to B **do** ▷ B = number of minibatches
4: $\mathbf{w} \leftarrow \mathbf{w} - \eta\, \hat{g}_b^{(t)}$

where

$$\hat{g}_b^{(t)} = \nabla_{\mathbf{w}} C_{\text{train},b}(\mathbf{w}) = \frac{1}{N_b} \sum_{n=1}^{N_b} \nabla_{\mathbf{w}} L_n$$

is the training cost of the b-th minibatch, with N_b denoting the number of examples in each minibatch and L_n the training loss of the n-th example in the minibatch at the t-th epoch.

In 2014, Kingma et al.[17] published the *adaptive moment estimation (Adam)* optimisation algorithm, which is an extension of SGD, specifically designed for non-convex optimisation

tasks. The main advantage of Adam is that it calculates individual learning rates for each parameter, and its popularity indicates that it converges much faster than any other optimisation algorithm for a majority of deep learning tasks. Specifically, each parameter changes with a different learning rate, which is adapted during training. Adam adapts the parameter learning rates based on the average of the first and second moment of the gradients. The k-th moment, m_k of a random variable A, refers to the expected value of the random variable raised to the k-th power. That is,$[m_k(A) = \mathbb{E}[A^k]$. Specifically, during training, the iterative algorithm calculates an exponential moving average of the gradient (first moment) as well as the squared gradient (second moment), where two parameters β_1 and β_2, pre-specified by the user, control the decay rates of these moving averages.

2.4 Deep Learning

In many real-world problems, the solution of a task may be highly complex. In machine learning (ML), we typically train a model using structured data represented in a design matrix X. This matrix contains features that are often the result of human expertise through *feature engineering*. However, a natural question arises: how can we be confident that the features extracted by a person are indeed the most informative for the ML task at hand? Moreover, traditional ML algorithms are not inherently designed to directly process unstructured data such as imagesor videos without substantial human intervention. These limitations motivated the development of *neural networks* (NNs). Neural networks are a class of ML architectures inspired by biological neurons, designed to recognise patterns in data. Conceptually, NNs can be viewed as linear models augmented with additional components called *hidden layers*, where the input data undergo a series of non-linear transformations. These transformations enable the network to capture and model complex relationships between inputs and outputs. Unlike classical models with a single weight vector, NNs involve fitting a large number of parameters across multiple layers of computation. However, training NNs is a more demanding process: they typically require large amounts of data, and their cost functions are no longer convex, which complicates optimisation. The term *deep learning* refers to neural networks with two or more hidden layers. In this chapter, we present the fundamental theory behind artificial neural networks (ANNs) and convolutional neural networks (CNNs).

2.4.1 Artificial Neural Network

The behaviour of biological neurons served as inspiration for the development of artificial neural networks (ANNs). Yet, just as airplanes were inspired by birds without the need to flap their wings [15], ANNs differ significantly from biological neurons in their actual functioning.

The perceptron, a fundamental form of ANN, was introduced in 1958 by Frank F. Rosenblatt [18]. Later, in 1969, Marvin Minsky and Seymour Papert published their influ-

ential book *Perceptrons* [19], which highlighted the severe limitations of perceptrons due to the computational resources available at the time. This critique marked the beginning of the period now known as the "AI Winter".

A simplified description of a biological neuron is that it consists of a cell body containing a nucleus and other components, multiple branch-like structures called *dendrites*, and a single long extension called the *axon*. The axon branches into smaller structures called *telodendria*, each ending in a *synaptic terminal*. These terminals connect to the dendrites of other neurons. In simple terms, a biological neuron receives input signals from many other neurons via its dendrites, processes the information in its cell body, and transmits the result through the axon to the synaptic terminals, which then relay the signal to connected neurons. An illustration of a biological neuron is shown in Figure 2.3

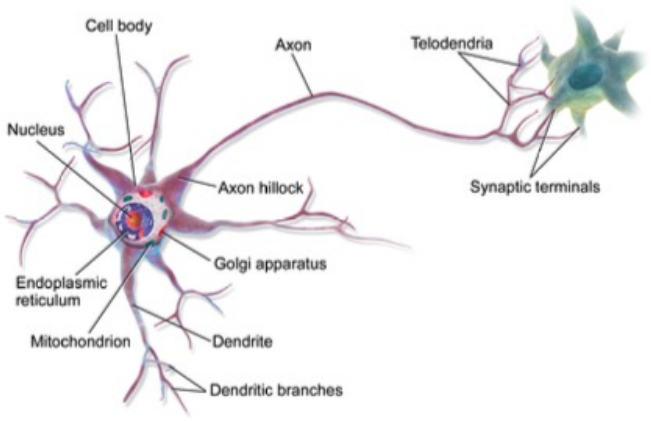


Figure 2.3: Illustration of biological neuron.

In analogy, for D features of a single training instance, a perceptron receives the input values, multiplies each by an associated weight, and adds a bias term. These values are combined and passed through an *activation function* $f(x)$, producing a single output value Figure 2.4.

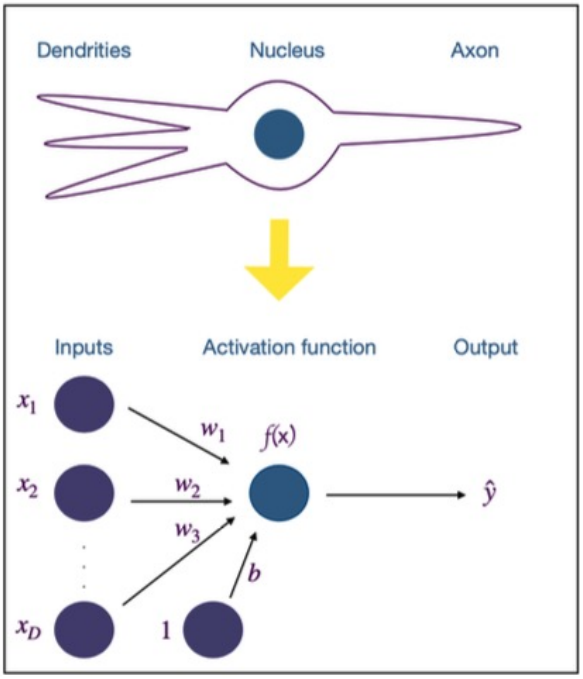


Figure 2.4: A simple biological neuron sketch in parallel with a single perceptron model.

Extending this idea, a more complex ANN architecture is formed by stacking multiple layers of perceptrons, resulting in a *multilayer perceptron* (MLP), as illustrated in Figure 2.5. In such networks, the outputs of one layer become the inputs of the next. This forward-only flow of information gives rise to the term *feedforward neural network*. The first layer is the *input layer*, which directly receives the data, while the last layer is the *output layer*, containing one or more neurons that represent the final predictions. All layers between them are called *hidden layers*. These hidden layers enable the network to capture complex interactions and patterns in the data.

In fully connected architectures, each neuron in a layer is connected to every neuron in the subsequent layer. While powerful, hidden layers are often difficult to interpret because of their dense connectivity and their distance from the directly observable inputs and outputs.

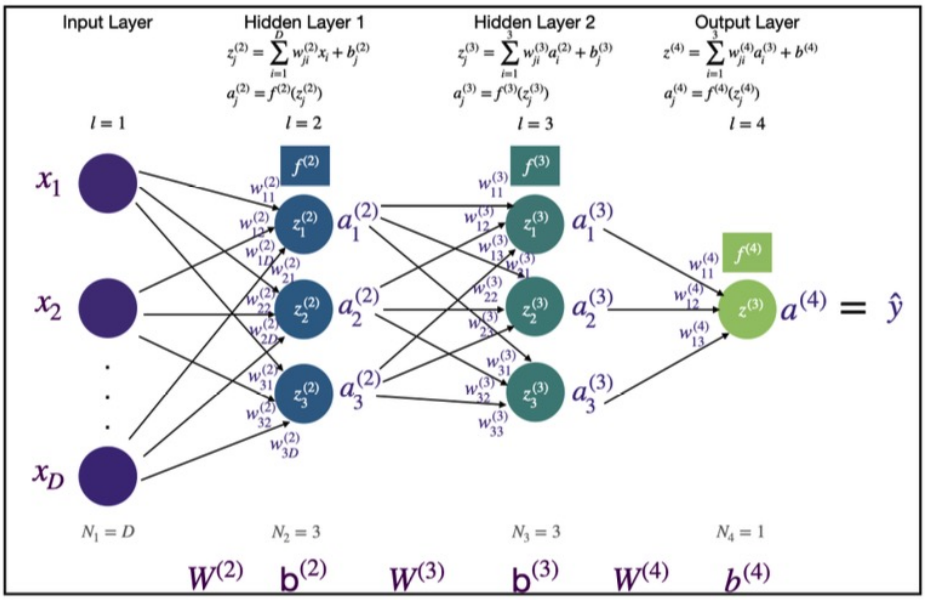


Figure 2.5: Fully connected NN with 2 hidden layers.

2.4.2 Activation functions

Activation functions are a fundamental component of neural networks (NNs). They determine the output behaviour of each neuron $z_j^{(l)}$ by applying non-linear transformations to the input $a_j^{(l-1)}$. Their most important role is introducing non-linearities into the network, enabling NNs to learn complex, non-linear mappings between inputs and outputs. Without activation functions, a network would be restricted to learning only linear relationships.

Since activation functions are applied to thousands or even millions of neurons during training, must also be computationally efficient. Over the years, extensive research has focused on their design and effectiveness [20]. Some common examples include:

- **Step Function:**

$$f(z) = \begin{cases} 0, & z \leq 0 \\ 1, & z > 0 \end{cases}$$

  This function produces outputs of either 0 or 1. It is very rigid, as small changes in z do not affect the output, making it unsuitable for most learning tasks.

- **Sigmoid Function:**

$$f(z) = \frac{1}{1 + e^{-z}}$$

Unlike the step function, the sigmoid is smooth and sensitive to small changes in z, providing a probabilistic interpretation. It is often used in the output layer for binary classification problems.

- **Rectified Linear Unit (ReLU):**

$$f(z) = \max(0, z)$$

ReLU is widely used in hidden layers because it helps address the vanishing gradient problem and generally provides strong performance in practice.

For binary classification tasks, the activation function in the output layer is typically the sigmoid function, as in logistic regression. For multi-class classification problems, the softmax function is used. The final output layer consists of C neurons, each corresponding to a possible class, and softmax converts their outputs into probabilities that sum to 1:

$$\text{Softmax}(z)_i = \frac{e^{z_i}}{\sum_{j=1}^{C} e^{z_j}}, \quad i = 1, \ldots, C.$$

This allows the model to output class probabilities, with the predicted class being the one with the highest probability. For example, if we use softmax for a classification task with three classes {Dog, Cat, Airplane}, the output could be {Dog: 0.2, Cat: 0.7, Airplane: 0.1}. In this case, the network predicts that the image is a *Cat* with 70% confidence.

2.4.3 Convolutional Neural Networks

While fully connected neural networks (ANNs) are capable of solving a variety of tasks, they are not the most effective choice for image data. In a standard ANN, the input layer is a flat, vertical layer of neurons corresponding to the features of a training sample. For example, an image of size 100×100 pixels would be flattened into $10,000$ input neurons. Even with a relatively small hidden layer of 100 neurons, this setup would require $1,000,000$ connections — corresponding to $1,000,000$ weights and 100 biases — for a single layer.

This exponential growth in the number of parameters, combined with the loss of spatial structure due to flattening, makes ANNs inefficient for image-related tasks. In flattened layers, the positional relationships of pixels are ignored, meaning that a pixel is treated the same regardless of its location. To overcome these limitations, *convolutional neural networks* (CNNs) were developed, inspired by studies of the brain's visual cortex, and have been used for image recognition since the 1980s [15]. CNNs are specialised neural network architectures particularly suited to image and video data, and they have proven effective in tasks such as image classification, object detection, and segmentation.

In essence, 2D CNNs employ *image kernels* (also called *filters* or *convolutional kernels*), which are small two-dimensional weight matrices applied across the entire image. The

convolution operation begins at the top-left corner of the image, where an element-wise multiplication between the kernel values and the corresponding pixel values is performed. The results are summed and combined with a bias, producing a single output value, which becomes the first neuron in the convolutional layer. To cover the entire image, the kernel is shifted, or *strided*, across the image. The stride length is specified by the user but cannot be less than one pixel.

When multiple kernels are applied, the result is a *convolutional layer*, where each kernel learns to detect different features. These layers form the fundamental building blocks of CNNs. One practical issue, however, is that as the kernel approaches the edges of an image, it lacks values to operate on, leading to information loss. A common solution is to use *zero-padding*, which pads the borders of the image with zeros, thereby preserving the spatial dimensions of the image during convolution.

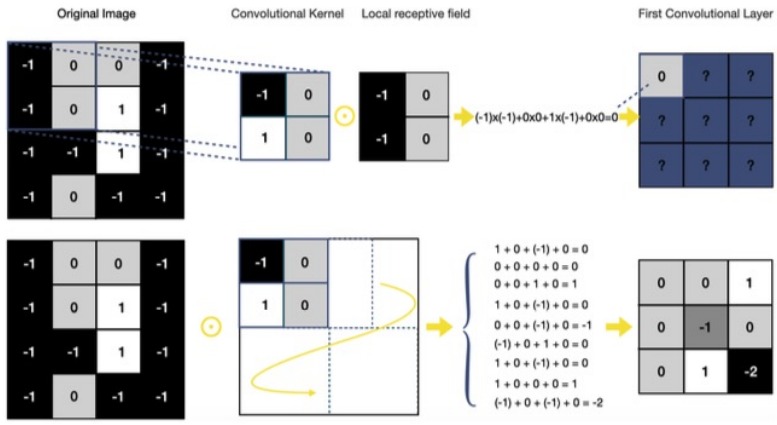


Figure 2.6: Single Convolutional Kernel applied into a 2D grayscale image with stride of one pixel.

A CNN architecture is based on three main components: local receptive fields, shared weights, and pooling.

Local Receptive Fields

Instead of creating full connections between the input layer and a convolutional layer, CNNs use connections restricted to *localised regions* of the input image. In other words, each neuron in a convolutional layer is connected only to a small, specific region of the input layer, as illustrated in Figure 2.6

This region in the input image is referred to as the *local receptive field* of the hidden neuron. The same principle generalises to any convolutional layer, where each neuron connects to its own local receptive field, as shown in Figure 2.7

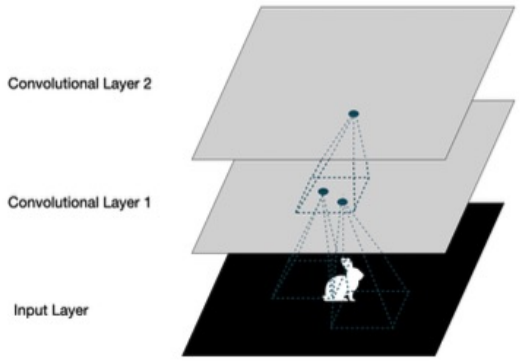


Figure 2.7: CNN layers with local receptive fields.

Shared Weights

As discussed earlier, each connection to a localised region of the input is formed by multiplying it with a *convolutional kernel*. This kernel is applied across all local receptive fields, meaning that each connection shares the same weights and a common bias.

For example, consider the case in [signle conv], where the input is a 4×4 image and a 2×2 convolutional kernel is applied. The output of a hidden neuron at position (j, k) is computed as:

$$\text{hidden}_{j,k} = f\left(b + \sum_{l=0}^{3}\sum_{h=0}^{3} w_{l,h}\, a_{j+l,k+h}\right),$$

where f is the activation function, b is the shared bias, $w_{l,h}$ are the elements of the convolutional kernel, and $a_{j+l,k+h}$ are the values from the corresponding local receptive field.

This formulation implies that all neurons in the first hidden layer detect the same feature, but at different locations within the input image. The result of applying a particular convolutional kernel is known as a *feature map*. Since each feature map corresponds to the detection of a single pattern, CNNs typically employ multiple feature maps in order to capture a variety of features from the input data.

The dimensions of a feature map depend on the size of the convolutional kernel, the size of the input image, the padding, and the stride. Specifically, the width and height of the feature map are given by:

$$\text{FM}_{\text{width}} = \frac{I_w - F_w + 2P}{S_w} + 1, \qquad \text{FM}_{\text{height}} = \frac{I_h - F_h + 2P}{S_h} + 1,$$

where:

- I_w, I_h are the input image's width and height,
- F_w, F_h are the convolutional kernel's width and height,
- P is the padding size,
- S_w, S_h are the stride in the width and height directions.

Pooling Layesrs

Another important component of CNNs is the use of *pooling layers*, which are typically placed immediately after convolutional layers. Even with local connections, convolutional layers can still produce a large number of parameters. The role of pooling is to reduce the dimensionality of the feature maps, thereby simplifying the information passed to the next layers.

A pooling layer condenses each feature map from the convolutional layer into a smaller, more manageable representation. The most common types are *max pooling* and *average pooling*. For example, with a 2×2 max-pooling layer, the output for each region it covers is a single scalar: the maximum value within that region. The intuition is that once a feature is detected, its precise location is less important than its approximate position relative to other features. Pooling therefore reduces redundancy and lowers the number of parameters, contributing to faster learning and improved generalisation.

In summary, the main differences between ANNs and CNNs lie in the structure of the input layer, the use of local (rather than full) connectivity, and the introduction of pooling layers to reduce parameters and accelerate training. Nevertheless, even in CNN architectures it is still necessary to *flatten* the neurons before the output layer. Typically, after the final convolutional and pooling layers, the neurons are vectorised into a fully connected layer, which links directly to the output layer, as illustrated in Figure 2.8.

Regarding the training process, the fundamental objective remains the same: to adjust the network's weights and biases using training instances, optimised through the back-propagation algorithm.

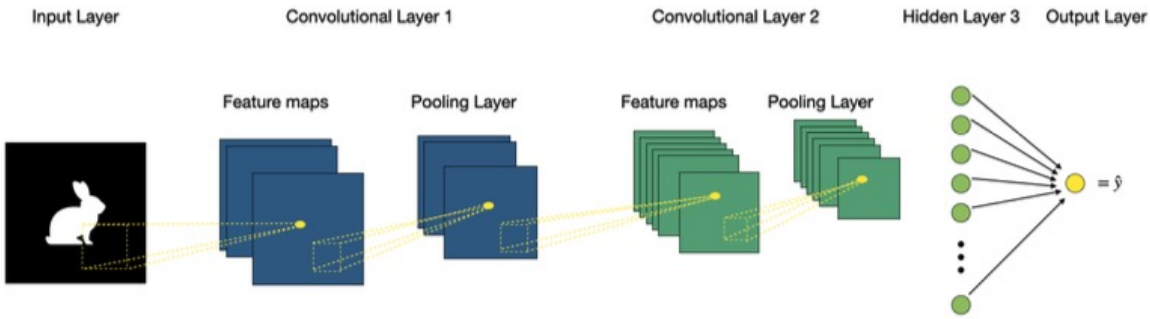


Figure 2.8: Example of a CNN architecture.

2.5 Privacy in Machine Learning

Machine learning (ML) systems depend fundamentally on access to data describing real individuals and their behaviors. When these data are used for training, the resulting model implicitly encodes statistical information about its sources. This raises a fundamental question: to what extent can an observer infer properties of the training data from the model itself? Privacy in machine learning therefore concerns the prevention of information leakage about individual data records, whether through model parameters, intermediate gradients, or observable outputs. A model is considered privacy-preserving when the inclusion or exclusion of a single training record does not significantly affect its output distribution [21, 2, 3].

From a theoretical standpoint, privacy can be formalized as a form of *algorithmic stability*. Let a learning algorithm A map a dataset $D = \{(x_i, y_i)\}_{i=1}^{N}$ to a trained model $M_\theta = A(D)$. Two datasets D and D' are called adjacent if they differ by one individual record. If, for every measurable subset S of possible outputs,

$$\Pr[A(D) \in S] \leq e^{\varepsilon} \Pr[A(D') \in S] + \delta,$$

then A satisfies (ε, δ)-*differential privacy* (DP) [21]. This property guarantees that the model's behavior remains nearly unchanged by the presence or absence of any single data point. Privacy can thus be interpreted as robustness to data perturbations, and a stable learning process tends to generalize better while leaking less.

However, most practical ML models are not inherently stable. Complex neural networks, trained via stochastic gradient descent on high-dimensional data, can memorize specific samples, particularly when training continues past the point of generalization. This memorization manifests in measurable behavioral differences: a model often predicts its training samples with higher confidence, lower loss, or smaller input-perturbation distances than unseen samples [22, 7]. Such discrepancies form the foundation of a broad class of *privacy attacks* that exploit model behavior to infer hidden properties of the training data.

Privacy concerns in ML therefore go beyond data confidentiality or access control. Even if a model and its parameters are securely stored, *information-theoretic leakage* can occur through legitimate queries. A model may be cryptographically secure yet statistically transparent. This distinguishes privacy from conventional security, because the adversary does not need to breach the system but can infer sensitive information simply by analyzing outputs or gradients.

The implications of these risks are both technical and societal. From a technical perspective, they expose the fundamental tension between utility and privacy. A model that completely obscures training influence is useless, whereas one that achieves high accuracy often memorizes individual samples. From a societal perspective, privacy underpins trust: individuals who contribute data to medical, financial, or behavioral datasets expect that their participation will not be detectable or reconstructable. Under the European Union's

General Data Protection Regulation (GDPR), any model that allows re-identification or membership inference may itself constitute personal data [8], extending regulatory obligations such as consent, transparency, and erasure rights to trained models.

In sum, privacy in machine learning refers to controlling what can be inferred about specific training data from a model's internal state or observable behavior. It is violated whenever a model's outputs reveal, even indirectly, the inclusion of individual records or sensitive patterns derived from them. Formally, this leakage corresponds to instability in the learning algorithm, and empirically it manifests as overfitting and memorization. Modern research therefore approaches privacy not merely as a data-protection problem but as a stability and generalization property of the learning process itself, motivating formal frameworks such as differential privacy and empirical analyses through membership inference attacks.

2.5.1 Privacy Threats in Machine Learning

Machine learning models are designed to identify patterns in data and generalize beyond the samples used during training. However, their ability to memorize fine-grained details can lead to the unintended exposure of sensitive information. Privacy attacks exploit this property by probing a trained model to infer private facts about its training data, without requiring direct access to the original dataset. These attacks differ from classical security breaches: instead of stealing stored data, they extract knowledge from the model's learned parameters or observable behavior [3].

At their core, such attacks rely on differences between how a model behaves on data it has seen during training and on data it encounters for the first time. Overfitted or unstable models tend to respond to familiar inputs with higher confidence or smaller prediction uncertainty, creating subtle but exploitable signals. An adversary can use these differences to infer information about individual data records or population characteristics, thereby violating the intended confidentiality of the training process.

Membership Inference Attacks

Membership Inference Attacks (MIAs) aim to determine whether a specific data record was included in the training dataset of a model. Given a model f_θ and a sample (x, y), the adversary's goal is to infer whether (x, y) belongs to the set of training examples. This decision is based on the model's observable responses, such as its output probabilities, prediction loss, or decision consistency under small input perturbations.

A successful membership inference indicates that the model's behavior differs between training and unseen data, meaning that the model has retained information specific to its training examples. The consequences can be severe: in medical or financial contexts, confirming that an individual's record was part of a training set may reveal private attributes about that person, such as a disease diagnosis or participation in a sensitive study. Because membership inference directly quantifies how much influence a single record has

on a model's behavior, it has become a standard metric for evaluating privacy risk in machine learning [2, 22, 7].

Model Inversion Attacks

Model Inversion Attacks target the reconstruction of input features or other sensitive attributes from a trained model's outputs. Instead of deciding whether a record is present, the adversary attempts to infer unknown characteristics of a record given partial information and access to model predictions. For example, a model that predicts disease likelihood might inadvertently reveal genetic or demographic traits of its training participants. By iteratively optimizing inputs to maximize the model's confidence in a specific output class, an attacker can approximate the underlying features that the model associates with that class, thereby revealing sensitive details about the data distribution [23].

Property and Attribute Inference

Property or Attribute Inference Attacks seek to uncover global characteristics of the training dataset rather than individual samples. The adversary's goal is to determine whether the data used for training possess a certain statistical property, such as the proportion of individuals from a specific demographic group or the presence of a sensitive label across the dataset. These attacks were first demonstrated in centralized learning settings, where an adversary could infer hidden properties of the dataset by analyzing model parameters [24], and were later extended to collaborative and federated learning, where shared gradients can reveal population-level statistics. [25].

Training Data Extraction Attacks

Training Data Extraction Attacks aim to recover explicit content from the training dataset by exploiting memorization in large, overparameterized models. This threat is particularly relevant for large language models, where a small fraction of parameters may store verbatim text fragments or identifiers from the data used for training. When carefully prompted, the model can reproduce these fragments, such as names, addresses, or passages, effectively leaking sensitive information [26]. Unlike membership inference, which determines inclusion, extraction attacks recover the actual data content.

Privacy attacks in machine learning vary in scope, but they share a common foundation: all exploit the instability of learning algorithms and the tendency of models to overfit. Membership inference measures whether a model treats training examples differently from unseen ones, model inversion reconstructs representative features of the input space, property inference uncovers statistical traits of the dataset, and extraction retrieves memorized content directly. Collectively, these attacks demonstrate that modern ML models can

reveal private information even when trained and deployed under standard security practices. Understanding these vulnerabilities is essential before discussing defense strategies such as regularization, confidence masking, and differential privacy, which aim to reduce or mathematically bound the leakage of information from trained models.

2.5.2 Defenses in Machine Learning

As privacy attacks reveal the ability of machine learning models to memorize and expose training information, various defensive strategies have been developed to reduce or bound this leakage. These defenses can be broadly categorized into *empirical approaches*, which rely on improving generalization and reducing overfitting, and *formal approaches*, which offer provable privacy guarantees grounded in mathematical definitions.

Empirical Defenses

Empirical defenses aim to make a model's responses to training and unseen data statistically indistinguishable. Since membership inference success correlates strongly with overfitting, methods that enhance generalization often provide partial mitigation against privacy leakage[22, 4].

A common empirical strategy is *regularization*, which constrains model parameters to prevent over-complex fitting. Typical examples include L_1 and L_2 weight penalties, dropout layers, and early stopping, where training halts once the validation error stops improving. These methods discourage memorization of rare or outlier samples, thereby narrowing the gap between training and test performance. However, their privacy effect is heuristic: they reduce the likelihood of leakage but do not provide any quantifiable privacy bound.

Another group of empirical techniques focuses on controlling the information revealed through model outputs. Prediction confidences can be *sanitized* by adding small random perturbations or by returning only the top-k predicted classes. Adversarial regularization and confidence masking further encourage models to produce similar output distributions for members and non-members[27]. Although such methods can hinder confidence-based membership inference, they remain vulnerable to attacks that exploit decision boundaries or input perturbations, such as label-only or boundary-based MIAs[5]. For this reason, empirical defenses are best viewed as mitigations rather than strict privacy guarantees.

Differential Privacy as a Formal Guarantee

While empirical techniques improve robustness through better generalization, they lack formal assurances about the information contained in the trained model. *Differential Privacy* (DP) provides a theoretical foundation for privacy [21] that quantifies and limits the effect of any single data record on the algorithm's output. A randomized training algorithm A satisfies (ε, δ)-differential privacy if, for any two datasets D and D' differing in

one record and for all subsets S of possible outputs,

$$\Pr[A(D) \in S] \leq e^{\varepsilon} \Pr[A(D') \in S] + \delta.$$

In this formulation, ε (epsilon) represents the *privacy budget*, which measures how much the output distribution may change due to the inclusion or exclusion of a single data point. Smaller ε values indicate stronger privacy. The parameter δ (delta) accounts for a negligible probability that the guarantee might not hold, setting δ close to zero yields stronger protection. In practice, differential privacy is implemented by adding random noise to sensitive computations, such as model gradients or aggregated statistics, thereby masking the influence of individual data points.

The amount of noise added depends on the sensitivity of the underlying function and on the desired privacy parameters (ε, δ). Higher noise levels provide stronger privacy but can reduce model accuracy, introducing the classic *privacy–utility trade-off* [28]. Differential privacy thus formalizes the concept of "privacy as stability": the presence or absence of any single record cannot significantly affect the learned model, limiting the success of attacks such as membership inference.

Other Privacy-Enhancing Frameworks

Additional privacy-preserving paradigms address information exposure from a systems perspective. *Federated learning* allows multiple participants to collaboratively train a model without sharing raw data instead, local models communicate gradient updates to a central server. While this reduces direct data transfer, shared gradients can still leak information about local datasets [25] and are often combined with differential privacy or encryption[29, 30]. Other frameworks, including *secure aggregation*, *homomorphic encryption*, and *multi-party computation*, protect intermediate values during distributed training but impose significant computational overhead and do not fully prevent statistical leakage.

Defending against privacy attacks requires balancing generalization performance, formal guarantees, and computational feasibility. Regularization, early stopping, and confidence masking offer lightweight, easily implemented mitigations but cannot provide provable security. Differential privacy, in contrast, defines an explicit and quantifiable bound on information leakage through the parameters ε and δ, independent of the adversary's capabilities. Although the addition of noise inevitably reduces model accuracy, DP remains the most reliable framework for guaranteeing that privacy loss stays within known limits. The following chapters evaluate the effectiveness of these defenses empirically by measuring how such techniques influence susceptibility to membership inference attacks.

2.5.3 Evaluating Privacy Leakage in Machine Learning

The degree to which a model exposes information about its training data can be quantified empirically through *privacy risk metrics*. These metrics assess how well an adversary can discriminate between training samples (*members*) and unseen samples (*non-members*) based on the model's observable behavior. Evaluating privacy leakage therefore becomes a problem of binary classification, where the attack model's predictions are compared against the ground-truth membership status of data points [2, 22, 3].

ROC Curve and AUC.

The most widely adopted framework for this evaluation is the *Receiver Operating Characteristic* (ROC) analysis. The ROC curve plots the *True Positive Rate* (TPR)—the fraction of correctly identified members—against the *False Positive Rate* (FPR)—the fraction of non-members incorrectly classified as members—as the attack decision threshold varies. The overall area under this curve, known as the *Area Under the Curve* (AUC), provides a threshold-independent measure of attack effectiveness. An AUC of 0.5 corresponds to random guessing, whereas an AUC of 1.0 indicates a perfect separation between members and non-members. In practice, higher AUC values signify that the target model exhibits more distinguishable behavior between its training and unseen data, and thus greater privacy leakage.

The ROC/AUC framework offers a convenient summary of the aggregate attack performance across thresholds. However, as Carlini et al. [7] argue, it fails to reflect the *worst-case* privacy loss. Average-case metrics such as AUC might appear benign even if an attack can confidently identify a small subset of individuals with near certainty. From a privacy and regulatory standpoint, even a handful of such disclosures constitutes a critical failure.

TPR at Low False Positive Rates.

To better capture these high-confidence breaches, Carlini et al. [7] introduced the use of *True Positive Rate at low False Positive Rates* (TPR@FPR). This metric evaluates the proportion of correctly identified members when the number of false positives is constrained to a very small percentage of non-members, typically 1% or 0.1%. The intuition mirrors that of other security disciplines, such as intrusion detection, where systems are required to detect genuine threats with negligible false alarms. A high TPR at an FPR of 0.1% implies that an adversary can reveal at least a few training records with overwhelming certainty, even if the average attack accuracy appears low.

From a mathematical perspective, if $f(x)$ denotes the attack score assigned to a sample x, then

$$\text{TPR@}\alpha = \frac{|\{x \in D_{\text{train}} : f(x) \geq \tau_\alpha\}|}{|D_{\text{train}}|}, \quad \text{where } \tau_\alpha = \min\{\tau : \text{FPR}(\tau) \leq \alpha\}.$$

This formulation explicitly constrains the acceptable false-positive rate α and evaluates the adversary's recall within that security boundary.

Interpretation and Use in Practice.

Both AUC and TPR@FPR metrics are complementary rather than competing. The AUC provides an overall view of attack separability, which is useful for model-to-model or defense comparisons. In contrast, low-FPR TPR focuses on the operational risk, the ability of an attacker to extract verifiable private information under realistic constraints. Recent empirical studies and auditing tools, such as *ML Privacy Meter* [8], adopt both metrics to generate privacy risk reports for deployed models. In such tools, the ROC curve provides an overview of the model's global privacy posture, while the TPR@1% and TPR@0.1% values indicate whether there exist outlier individuals at elevated exposure risk.

Overall, privacy evaluation metrics translate abstract leakage into quantifiable risk. A model that exhibits high AUC or elevated TPR at low FPRs is considered *privacy-vulnerable*. Conversely, defenses such as regularization or differential privacy aim to minimize these indicators, aligning the model's responses to members and non-members and thereby reducing the adversary's advantage.

Chapter 3

Related Work on Membership Inference Attacks

Membership Inference Attacks (MIAs) have undergone substantial methodological evolution since their formal introduction. Early approaches relied on extensive adversarial knowledge, computational resources, and strong assumptions about data distribution. Over time, subsequent research has relaxed these requirements, demonstrating that effective attacks can succeed even when the adversary has limited access, such as observing only predicted class labels. Recent work has further expanded the MIA paradigm beyond classical classification to include unsupervised learning, regression, and federated architectures, underscoring the pervasiveness of privacy leakage across diverse machine learning settings.

This chapter adopts a chronological structure to trace the progression of MIA methodologies. Each approach is analyzed in terms of its technical contribution, level of adversarial access, and underlying assumptions. By situating these attacks within a unified timeline, the chapter highlights how MIAs have evolved from theoretical demonstrations to practical privacy threats. It concludes with a synthesis that categorizes attack variants by their operational requirements and the privacy vulnerabilities they expose, offering a conceptual framework for evaluating defenses in later chapters.

3.1 The Emergence of Membership Inference

Shokri et al. [2] laid the groundwork for Membership Inference Attack (MIA) research by introducing a structured and modular pipeline. Their method trains multiple *shadow models* that approximate the behavior of a target model by being trained on datasets drawn from the same or similar distributions. The outputs of these shadow models are then used to construct a new dataset labeled according to whether each data point was part of the shadow model's training set or not. This labeled dataset serves as input to a separate *inference model*, which acts as the final attacker. Its task is to predict whether a given

sample was part of the target model's training data, based solely on the observable output behavior of the target, such as prediction confidence vectors.

This framework formalized membership inference as a supervised learning task in its own right, where the attacker learns to distinguish members from non-members by analyzing the target model's response patterns. Although highly effective, the technique assumes black-box access to the model's outputs, knowledge of the training data distribution, and enough computational resources to train multiple auxiliary models. Still, the architecture proved robust and adaptable, shaping the design of both later attacks and defensive strategies. It introduced the central adversarial intuition that continues to underpin MIA research: models behave differently on data they have seen during training than on unseen data.

The novelty of this work also lies in its abstraction. The methodology is not bound to any specific model architecture or dataset, making it applicable to diverse tasks such as image or text classification. As a result, Shokri et al.'s approach became a reference framework for subsequent studies, a blueprint for constructing attacks and a benchmark for evaluating defenses. It marked the formal emergence of membership inference as a distinct category of privacy attacks, establishing the foundation upon which all later refinements were built.

3.2 Lowering the Barrier: Metric-Based and Loss-Based Simplifications

Following the foundational work of Shokri et al. [2], subsequent studies demonstrated that membership inference could be achieved with far fewer assumptions and computational requirements. This shift marked a move from complex shadow-model pipelines toward simpler, more generalizable strategies that exploit intrinsic signals of overfitting.

3.2.1 Salem et al. (2018): Metric-Based Attacks and the ML-Leaks Paradigm

Salem et al. [4] introduced a series of simplifications to the original shadow model framework, forming what later became known as the *ML-Leaks* paradigm. They showed that effective membership inference attacks could be mounted with limited auxiliary data, relying on only a single shadow model or even none at all. By using direct statistical metrics such as prediction confidence, entropy, or top-k probability, their approach inferred membership status without the need for a trained inference network.

The authors also proposed several relaxed shadow-model configurations that deviated from the original setup, including mismatched training distributions, heterogeneous architectures, and transfer learning from related domains. Despite these relaxations, the attacks maintained strong predictive performance, demonstrating that the core leakage signal is not dependent on architectural alignment but on general overfitting behavior. This work opened the door to scalable membership audits of practical Machine Learning as a Service

(MLaaS) APIs, where adversaries have limited visibility into training data and minimal resources for replication. It established that privacy leakage is a general property of over-parameterized models rather than a by-product of Shokri's experimental design.

3.2.2 Yeom et al. (2018): Loss-Based Thresholding

Yeom et al. [22] introduced one of the earliest white-box membership inference attacks that completely eliminated the need for shadow models. Their approach relies on the observation that overfitted models tend to assign lower loss values to data points seen during training compared to unseen samples. By computing the model's per-sample loss and comparing it against a fixed threshold, an adversary can predict whether a given example was part of the training dataset.

This loss-based thresholding method is conceptually simple yet empirically effective. It formalized the connection between overfitting, generalization gap, and membership leakage, showing that internal model metrics can serve as reliable privacy indicators. Moreover, it influenced the development of later theoretical analyses linking model loss distributions to privacy risk, positioning loss-based attacks as a baseline for evaluating both empirical and differentially private defenses.

3.3 White-Box and Gradient-Based Attacks

As machine learning systems grew in complexity and collaborative learning architectures emerged, researchers began to investigate attacks under stronger adversarial assumptions, where internal model states are partially or fully exposed. This new generation of *white-box* attacks leveraged gradients, parameters, and activations to design more powerful and fine-grained inference techniques.

3.3.1 Nasr et al. (2019): Privacy Analysis Through White-Box Inference

Nasr et al. [27] conducted one of the first systematic studies of white-box membership inference attacks, demonstrating that access to internal model information substantially amplifies privacy risk. Their proposed attack model aggregates multiple features including prediction confidence, loss, gradients of the loss with respect to parameters, and intermediate layer activations to train a binary inference classifier distinguishing members from non-members.

Their evaluation revealed that federated learning systems are particularly susceptible, as periodic gradient updates shared among participants expose rich leakage channels. An adversary can exploit these updates passively by monitoring shared parameters, or actively by manipulating local model behavior to enhance separability between member and non-member samples. This work firmly established white-box MIAs as a credible

threat model in distributed learning environments and motivated subsequent defenses based on gradient perturbation and differential privacy.

3.3.2 Leino & Fredrikson (2020): Calibration and Memorization

Leino and Fredrikson [31] deepened the analysis of white-box privacy leakage by connecting it to model calibration and memorization dynamics. They showed that poorly calibrated models, where predicted confidence diverges from actual accuracy, tend to leak membership information more readily. Their methodology used layer-wise activations and confidence statistics to infer membership, revealing how training-time overconfidence correlates with memorization of individual samples.

By introducing calibration curves and per-layer metrics into the inference process, the authors provided a more diagnostic perspective on privacy risk, treating it as a measurable artifact of learning dynamics rather than a purely post hoc outcome. Their findings also informed a new class of defenses, such as temperature scaling and early stopping, which improve calibration and thereby reduce leakage indirectly. This study bridged the conceptual gap between memorization, generalization, and privacy, positioning calibration as both a vulnerability metric and a defensive lens.

3.4 Label-Only, Contrastive, and Memorization-Aware Attacks

As machine learning applications diversified beyond classical supervised learning, new membership inference techniques emerged that challenged prior assumptions about what information was required to compromise privacy. Recent research has shown that even models providing minimal outputs or trained without labels can still leak training information through their internal representations or behavioral consistency.

3.4.1 Liu et al. (2021): Contrastive Representation Leakage

Liu et al. [32] introduced *EncoderMI* in their paper *Membership Inference Attack against Generative Encoders*, targeting self-supervised learning frameworks that rely on contrastive objectives. Their work demonstrated that models trained without explicit labels can still memorize training samples in ways that enable membership inference.

EncoderMI measures the similarity between latent representations of test samples and those of known training samples. The assumption is that training data yield embeddings that are more tightly clustered or more similar to each other than embeddings from unseen examples. By computing cosine similarity and fitting statistical models to these distance distributions, the attacker distinguishes members from non-members, even in the absence of logits or labels.

This contribution was significant because it extended membership inference beyond supervised contexts to representation learning systems, which are increasingly used in

modern applications. EncoderMI showed that even models trained solely on contrastive objectives, such as MoCo, BYOL, or SimSiam, remain vulnerable to privacy leakage. This finding invalidated the assumption that supervision is necessary for memorization-based attacks.

Its implications are far-reaching. In contemporary machine learning pipelines, large foundation models and reusable encoders are trained on vast datasets and later fine-tuned or integrated into downstream systems without transparency about the original training data. EncoderMI revealed that such encoders can encode and leak membership information through their learned feature spaces, underscoring the need for privacy-preserving mechanisms that extend beyond the output layer.

3.4.2 Label-Only Attacks

The emergence of *label-only* membership inference attacks represented a turning point in the privacy threat landscape. These attacks demonstrated that even models exposing only a final class label, with no confidence scores, can still leak sensitive information. They exploit how models respond to input perturbations, using the stability or robustness of predicted labels as a signal of membership. Three major studies define the evolution of this attack family.

Choquette-Choo et al. [33] introduced the first label-only attack achieving accuracy comparable to traditional confidence-based approaches. Their insight was that training samples produce more stable predictions under perturbations than unseen samples. By repeatedly querying the target model with transformed versions of an input and observing label consistency, they inferred membership without access to probability vectors. This multi-query framework effectively simulated confidence information through repeated decision outcomes. However, it required thousands of queries per input, increasing computational cost and detectability in practical deployments.

Li and Zhang [34] later proposed two practical label-only methodologies under decision-only constraints: the *transfer attack* and the *boundary attack*. The transfer attack constructs a shadow dataset labeled by the target model to train a local surrogate, enabling conventional score-based attacks through proxy modeling. The boundary attack, by contrast, operates directly on the target model by introducing adversarial perturbations. Its core assumption is that members lie farther from the decision boundary, requiring larger perturbations to alter their predicted label. The resulting perturbation magnitude becomes a surrogate for membership likelihood. These methods demonstrated that even under minimal access assumptions, adversaries can exploit geometric properties of the decision space to infer membership.

Building on these developments, Peng et al. [6] proposed *OSLO* (One-Shot Label-Only), a high-precision attack representing the current state of the art. OSLO overcomes the high query complexity and low precision of previous methods by combining transfer-based adversarial ideas with statistical hypothesis testing. It trains multiple surrogate models

on public data to generate targeted adversarial examples for each input. If the example fails to change the target model's decision, the input is predicted as a member otherwise, it is classified as a non-member. OSLO achieves strong results using a single query per sample, with true positive rates up to 22 times higher than prior label-only methods under low false positive constraints (FPR 0.1%). Its adaptive, per-sample perturbation budgets make it both efficient and precise.

Collectively, label-only attacks demonstrate that even the most constrained model interfaces are not immune to privacy risks. They reveal that robustness, long regarded as a desirable property, can itself become an exploitable feature, as members tend to exhibit higher prediction stability. These results challenge the sufficiency of confidence masking or output truncation as defensive strategies.

3.4.3 Carlini et al. (2022): Membership Inference from First Principles

Carlini et al. [7] revisited the foundations of membership inference and proposed a rigorous statistical framework for evaluating privacy risk. They observed that many previous studies measured attack success through aggregate accuracy or AUC, which can be misleading when false-positive rates are high. To address this, they introduced the *Likelihood Ratio Attack* (LiRA), a method that interprets membership inference as a hypothesis-testing problem. Rather than treating attack performance as a binary classification outcome, LiRA compares how likely a model's outputs are under two competing hypotheses: that a sample was included in training or that it was not. By estimating these likelihoods from ensembles of shadow models, the attack produces a per-sample privacy score grounded in statistical theory.

This reformulation established a principled way to quantify privacy leakage and enabled fair comparison of attacks across architectures and datasets. It also revealed that the true privacy risk often lies in rare, atypical samples, not simply in overall overfitting. Importantly, Carlini et al. emphasized that privacy evaluations should focus on realistic low–false-positive settings, where even a few incorrect inferences can have severe implications. Their framework has since become a cornerstone of empirical privacy auditing, providing the theoretical basis for subsequent work that connects membership inference to memorization dynamics, most notably the analysis by Li et al. (2024).

3.4.4 Li et al. (2024): Reassessing Privacy Through Memorization

Building upon the formal framework introduced by Carlini et al. [7], Li et al. [35] reinterpreted membership inference through the concept of *memorization*. Rather than evaluating privacy leakage solely through statistical metrics such as accuracy or likelihood ratios, they argued that the true risk stems from how much influence each individual sample exerts on the trained model. A model that depends strongly on a specific record is said to have *memorized* it, making that record inherently more vulnerable to inference.

Using this lens, the authors analyzed a broad spectrum of membership inference attacks, including confidence-, loss-, and entropy-based methods, as well as learned attack classifiers. They found that many of these approaches correlate only weakly with genuine memorization. In practice, they often identify easily classifiable, low-risk members while failing to detect the atypical or outlier samples that the model has actually memorized. This inconsistency highlights a crucial insight: privacy leakage cannot be inferred from overall accuracy or generalization gap alone, but must be understood as a sample-specific phenomenon.

To quantify this relationship, Li et al. compared existing attacks against empirically measured memorization across multiple architectures and datasets. They showed that the most consistent results with true memorization were achieved by likelihood-based methods such as LiRA, confirming that hypothesis-testing approaches better capture the causal link between learning dynamics and privacy exposure. However, even these methods exhibited variation across datasets and defense settings, suggesting that memorization is a richer, multidimensional construct rather than a single numerical score.

Finally, the study examined how data enhancement techniques, such as data augmentation and adversarial training, affect privacy when viewed through memorization. Contrary to common expectations, these strategies do not uniformly reduce risk. Data augmentation tends to redistribute memorization by lowering it for typical samples while increasing it for rare or complex ones, whereas adversarial training can either mitigate or amplify memorization depending on its configuration. These findings reveal that improvements in robustness or generalization do not necessarily translate into stronger privacy.

Overall, Li et al. established memorization as a unifying framework for understanding membership inference and evaluating defenses. By linking individual-sample influence to empirical privacy risk, their work provides a practical foundation for future privacy auditing tools and for developing training procedures that explicitly account for how models remember, and potentially expose, the data on which they learn.

3.5 Summary and Conceptual Taxonomy

The evolution of membership inference reflects a steady broadening of both attack capability and theoretical understanding. Early frameworks required extensive auxiliary data and multiple shadow models, while modern approaches succeed with minimal information, including single-label outputs. In parallel, the field's analytical focus has shifted from heuristic metrics to statistically and causally grounded formulations.

Table 3.1 summarizes the major classes of MIAs by adversarial access, methodological principle, and key contribution.

In summary, membership inference research has progressed from engineering attacks to developing quantitative privacy diagnostics grounded in theory. The field now offers both a taxonomy of adversarial access levels and a unifying perspective in which privacy risk emerges from the statistical and memorization properties of the learning process itself.

Table 3.1: Taxonomy of membership inference attacks by access level and methodological principle.

Access Level	Representative Works	Core Insights and Contributions
Black-box (Confidence)	Shokri et al. (2017); Salem et al. (2018)	Shadow-model framework formalizing MIAs; simplified metric-based attacks enabling practical MLaaS audits.
White-box (Gradient)	Nasr et al. (2019); Leino & Fredrikson (2020)	Exploitation of internal activations and gradients; identification of calibration and memorization as drivers of leakage.
Representation-Level (Contrastive)	Liu et al. (2021)	Demonstrated leakage from latent encoders in self-supervised and contrastive learning models.
Label-Only (Decision)	Choquette-Choo et al. (2021); Li & Zhang (2021); Peng et al. (2024)	Showed that even models exposing only class labels leak privacy; introduced perturbation-based stability metrics.
Formal / Statistical (Auditing)	Carlini et al. (2022); Li et al. (2024)	Established likelihood- and memorization-based frameworks for principled, quantitative privacy evaluation.

Chapter 4

Methodology

This chapter presents the experimental methodology used to evaluate how training configurations influence both model utility and privacy leakage under different adversarial settings. Three types of membership inference attacks (MIAs) are implemented, representing a progression in attacker capability and realism:

1. **Score-based attack:** classical confidence-based MIA applied across multiple datasets.
2. **One-shadow model attack:** learning-based MIA using one auxiliary model [4] trained on disjoint data .
3. **Transfer (label-only) attack:** strongest black-box variant using label-only access and surrogate training [5] .

.

4.1 Experimental scope and principles

The primary objective is to quantify the relationship between generalisation and privacy leakage in vision models trained with and without differential privacy. All experiments are implemented in **PyTorch** with **Opacus** for DP-SGD training. Random seeds and deterministic backends are fixed to ensure reproducibility. Training and evaluation are performed on a single GPU when available (`device="cuda"`).

4.1.1 Training regimes

Each dataset or attack configuration follows one of four possible regimes:

1. **Baseline:** non-private, no explicit regularisation.
2. **Regularised:** dropout and/or early stopping (ES) to mitigate overfitting.

3. **DP:** training with differentially private stochastic gradient descent (DP-SGD).

4. **DP+Regularised:** combination of DP-SGD and regularisation .

The DP mechanism follows the standard per-sample gradient clipping and Gaussian noise addition implemented in Opacus. Noise multipliers $\sigma \in \{0.5, 0.8, 1.0, 1.2, 1.5\}$ are tested, with clipping norm C=1.0 and $\delta \approx 1/N_{\text{train}}$. All models use batch size 256 and learning rate 10^{-3}. Training proceeds for up to 100 epochs, with early stopping (patience = 10) when enabled.

4.1.2 Evaluation metrics

Each attack is evaluated using three privacy metrics:

$$\text{(i) AUC,} \qquad \text{(ii) TPR@0.1\% (FPR} = 0.001\text{)}, \qquad \text{(iii) TPR@1\% (FPR} = 0.01\text{)}.$$

The *Area Under the ROC Curve* (AUC) measures overall separability between members and non-members, while TPR@FPR values quantify attack success under strict false-positive constraints, following best practices in security evaluation [3, 26]. Model utility is measured as top-1 test accuracy, representing true generalisation to unseen data.

4.2 Score-based membership inference attack

4.2.1 Threat model

The score-based attack represents the classical black-box MIA of Shokri et al. [2]. The attacker can query the model f_θ and observe its softmax output vector. The assumption is that training samples yield higher predicted confidence than unseen samples, making the maximum softmax probability a useful membership indicator.

4.2.2 Attack formulation

For each queried sample x, the adversary computes:

$$s(x) = \max_y f_\theta(x)_y,$$

where $f_\theta(x)_y$ denotes the predicted probability for class y. Membership is inferred by thresholding $s(x)$; higher values imply stronger evidence of membership. Sweeping the threshold τ across all possible values produces the ROC curve, from which AUC and TPR@FPR are derived.

4.2.3 Datasets and preprocessing

This attack is applied to five visual datasets: CIFAR-10, CIFAR-100, OCTMNIST, RetinaMNIST, and PathMNIST. Each dataset is split into training, validation, and test partitions as shown in Table 4.1. A balanced evaluation set is created by combining equal numbers of member (training) and non-member (test) samples. All images are normalised to $[0, 1]$, and medical datasets are resized to 28×28 pixels. For transfer-learning runs, ImageNet normalisation and resizing to 224×224 are used.

Table 4.1: Dataset characteristics and membership-evaluation composition for the score-based attack.

Dataset	Domain	Input	Classes	Train/Val/Test	MIA (M/NM)
CIFAR-10	Natural images	3×32×32	10	45k / 5k / 10k	10k / 10k
CIFAR-100	Natural images	3×32×32	100	45k / 5k / 10k	10k / 10k
OCTMNIST	Retinal OCT scans	3×28×28	4	97k / 10k / 1k	5k / 5k
RetinaMNIST	Fundus camera	3×28×28	5	1.1k / 0.1k / 0.4k	0.4k / 0.4k
PathMNIST	Colon pathology	3×28×28	9	90k / 10k / 7.2k	7.2k / 7.2k

4.2.4 Architectures and training regimes

The architectures used for the score-based experiments are listed in Table **??**. Each model family supports regularisation (dropout, early stopping) and DP training through the Opacus engine. For DP-SGD configurations, BatchNorm layers are replaced with GroupNorm to maintain privacy accounting consistency. All models are trained for up to 100 epochs or until early stopping is triggered based on validation performance.

Table 4.2: Model families and corresponding training regimes for score-based MIA experiments.

Dataset	Model type	Regularization	DP mechanism
CIFAR-10	CNN (Tanh / LeakyReLU)	Dropout, ES	DP-SGD (Opacus)
CIFAR-100	ResNet-18 / WideResNet-28-10	Dropout, ES	DP-SGD (BN→GN)
OCTMNIST	ResNet-18 (28×28)	Dropout, ES	DP-SGD (BN→GN)
RetinaMNIST	ResNet-18 (28×28)	Dropout, ES	DP-SGD (BN→GN)
PathMNIST	ResNet-18 (28×28)	Dropout, ES	DP-SGD (BN→GN)

4.2.5 Evaluation

For each configuration, we compute the ROC curve, AUC, TPR@0.1%, and TPR@1%. These results serve as the baseline privacy benchmarks for comparison against the one-shadow and transfer attacks, which target CIFAR-10 more specifically.

4.3 One-shadow model membership inference attack (CIFAR-10)

4.3.1 Threat model

The one-shadow attack [4] assumes an adversary with black-box access to the target model and a separate dataset from the same distribution. The attacker trains a single shadow model g_ϕ to emulate the target's behaviour on member and non-member data, then learns an attack classifier to distinguish them. The adversary observes softmax outputs (class posterior probabilities) from the shadow model but never accesses the target's parameters.

4.3.2 Dataset split

CIFAR-10. The CIFAR-10 dataset (60,000 images, 6,000 per class) is split into two disjoint, class-balanced pools:

$$D_{\text{target}} = 30{,}000, \quad D_{\text{shadow}} = 30{,}000.$$

Within each pool we create stratified subsets:

$$\begin{aligned} D_{\text{target}}: &\quad 23{,}000 \text{ train},\ 2{,}000 \text{ val},\ 5{,}000 \text{ test}; \\ D_{\text{shadow}}: &\quad 14{,}000 \text{ train},\ 1{,}000 \text{ val},\ 15{,}000 \text{ out}. \end{aligned}$$

All splits preserve per-class balance and are mutually disjoint.

PathMNIST. PathMNIST (107,180 patches) is handled with a global concatenation of the original `train/val/test` partitions followed by a reproducible global shuffle (fixed RNG seed). Fixed, non-stratified counts are allocated as:

$$\begin{aligned} D_{\text{target}}: &\quad 40{,}552 \text{ train},\ 3{,}039 \text{ val},\ 10{,}000 \text{ test}; \\ D_{\text{shadow}}: &\quad 25{,}009 \text{ train},\ 1{,}786 \text{ val},\ 26{,}794 \text{ out}. \end{aligned}$$

These index sets are non-overlapping and cover the entire dataset. Because the split is not class-stratified, per-class counts may vary across splits; we keep this design to reflect realistic clinical imbalances.

4.3.3 Model architecture

Target and shadow models share the same lightweight `TanhCNN` template. For CIFAR-10 the network outputs 10 logits, for PathMNIST the same template is used with a 9-logit output head. Implementation details (two 3×3 conv layers with tanh + 2×2 max-pooling, a 128-unit FC, optional dropout controlled by p) follow the codebase; we omit further low-level specifics here for brevity. Dropout probability p is toggled to model Baseline (no dropout), Regularised, and DP+Regularised regimes.

4.3.4 Attack construction

For both datasets the attack pipeline is identical:

1. Train the shadow model on $D_{\text{shadow,train}}$ and validate on $D_{\text{shadow,val}}$.
2. For every sample in $D_{\text{shadow,train}}$ (members, label m=1) and $D_{\text{shadow,out}}$ (non-members, label m=0), query the trained shadow model and extract features: the top-3 predicted probabilities (sorted descending) and the predicted class label.
3. Assemble a binary attack dataset of feature vectors and membership labels $(\text{features}, m)$. Train a binary classifier (the *attack model*) on this dataset.
4. To evaluate leakage, query the target model on its own training set ($D_{\text{target,train}}$, true members) and on held-out non-members ($D_{\text{target,test}}$). Apply the trained attack model to these predictions to produce membership scores and compute evaluation metrics (ROC, AUC, and TPR at low FPR thresholds).

All choices (top-3 posteriors, use of predicted class, and the single-shadow workflow) match the canonical shadow model, black-box attack methodology [2, 4] used elsewhere in this work.

4.3.5 Training regimes and evaluation

The shadow and target models are trained under four regimes: **Baseline**, **Regularised** (dropout $p = 0.3$, early stopping), **DP** (noise multiplier $\sigma \in \{0.8, 1.2\}$), and **DP + Regularised** (DP-SGD with the same σ values combined with dropout and early stopping). Both shadow and target models are trained for up to 100 epochs, or until early stopping is triggered on the respective validation split. The attack model is trained without DP constraints.

Privacy leakage is quantified using Area Under the ROC Curve (AUC) and low-FPR diagnostics (TPR@0.1% and TPR@1%). Evaluation is performed on balanced membership evaluation sets: for CIFAR-10 we use 5,000 members and 5,000 non-members, for PathMNIST we use 10,000 members and 10,000 non-members. These metrics measure how effectively a learned discriminator recovers membership compared to simple threshold-based baselines, with special emphasis on performance at very low false-positive rates relevant to practical adversaries.

4.4 Transfer (label-only) membership inference attack (CIFAR-10)

4.4.1 Threat model

The transfer (label-only) attack [5] models the most restrictive black-box adversary. The attacker can query the target model f_θ but only observes the top-1 predicted class label,

not confidence scores or logits. The attacker also controls a large, disjoint dataset drawn from the same distribution. This auxiliary data is relabelled by querying the target model, and then used to train a surrogate model that imitates the target's decision boundary.

4.4.2 Dataset split

CIFAR-10. We explicitly separate a small "private" target model from a data-rich attacker. CIFAR-10 is partitioned as:

$$\begin{array}{lll} \text{Target:} & D_{\text{target}} = 3{,}000 & (100 \text{ train}, 100 \text{ val}, 100 \text{ test per class}), \\ \text{Shadow:} & D_{\text{shadow}} = 45{,}000 & (4{,}500 \text{ per class}). \end{array}$$

The remaining 15,000 samples (1,500 per class) are discarded. The evaluation pool for membership inference consists of 1,000 members (the target training set) and 1,000 non-members (the target test set), with 100 per class in each.

PathMNIST. We apply the same logic to PathMNIST (9-class histopathology patches). After concatenating the original `train/val/test` splits into a single indexed pool, we sample per class using fixed quotas:

$$\begin{array}{lll} \text{Target:} & D_{\text{target}} = 300 \text{ per class} & (100 \text{ train}, 100 \text{ val}, 100 \text{ test per class}), \\ \text{Shadow:} & D_{\text{shadow}} = 4{,}500 \text{ per class.} & \end{array}$$

For each class c, we first allocate 100 samples to $D_{\text{target,train}}$, 100 to $D_{\text{target,val}}$, and 100 to $D_{\text{target,test}}$, and then assign up to 4,500 further samples of that class to D_{shadow}. If a rare class does not have enough examples to satisfy the full 4,500 shadow quota, the shadow portion for that class is reduced accordingly. As in CIFAR-10, evaluation uses two balanced sets drawn from the target model only: members taken from $D_{\text{target,train}}$ (100 per class) and non-members from $D_{\text{target,test}}$ (100 per class), i.e. 1,000 vs. 1,000 when 10 classes are present and ≈ 900 vs. ≈ 900 in the 9-class case, subject to class availability.

All index sets are non-overlapping: no image used to train the target ever appears in the attacker's shadow pool.

4.4.3 Model architecture

Both the target model and the attacker's surrogate model use the same small `TanhCNN` used in the one-shadow setting. This network consists of two 3×3 convolutional layers with tanh activations and 2×2 max-pooling, followed by a fully connected layer of 128 units (again with tanh), optional dropout, and a final linear output layer. The only difference between datasets is the output dimensionality: 10 logits for CIFAR-10, 9 logits for PathMNIST. We intentionally keep the architecture simple and aligned across target and surrogate, so that attack success reflects the label-only pipeline rather than model capacity differences.

4.4.4 Attack procedure

The transfer (label-only) attack is carried out in three stages: *shadow dataset relabeling*, *shadow model training*, and *membership inference*.

Shadow dataset relabeling. The adversary possesses a shadow dataset D_{shadow} drawn from the same distribution as the target dataset D_{target}. To train a shadow model, the adversary first queries the target model,that acts as an oracle, f_θ on each $x \in D_{\text{shadow}}$ and records only the top-1 predicted label. The resulting relabelled pairs $(x, \hat{y} = \arg\max f_\theta(x))$ link the shadow dataset to the target's decision behaviour, enabling the next stage of model imitation.

Shadow model training. The adversary trains a shadow model S on the relabelled dataset $(x, \hat{y}) \in D_{\text{shadow}}$ using standard supervised learning with cross-entropy loss, SGD/Adam optimisation, and optional early stopping. A small validation split of D_{shadow} is used to tune hyperparameters and to later calibrate the membership threshold.

Membership inference. Given a candidate input x with ground-truth label y, the adversary computes the shadow model's cross-entropy loss:

$$\mathcal{L}_{\text{CE}}(x; S) = -\sum_{i=1}^{K} \mathbf{1}_{\{i=y\}} \log p_i(x; S),$$

where $p_i(x; S)$ is the predicted probability for class i, $\mathbf{1}_{\{i=y\}}$ is the one-hot encoding of the true label, and K is the number of classes. If $\mathcal{L}_{\text{CE}}(x; S)$ is smaller than a threshold τ (estimated from the shadow validation set), the sample is classified as a *member*; otherwise, it is considered a *non-member*. This decision rule is applied identically across both CIFAR-10 and PathMNIST, using their respective target models for relabeling and the same `TanhCNN` architecture (10 logits for CIFAR-10, 9 logits for PathMNIST).

4.4.5 Training regimes and evaluation

The target model is trained under four regimes: **Baseline**, **Regularised** (dropout $p = 0.3$), **DP** (noise multiplier $\sigma \in \{0.8, 1.2\}$), and **DP + Regularised** (DP-SGD with the same σ values combined with dropout). To intentionally induce overfitting and thus amplify potential membership signals, the target model is trained for 200 epochs, whereas the shadow model is trained for 100 epochs using the same optimisation settings. The Shadow model is always trained non-privately.

Privacy leakage is quantified using AUC and low-FPR diagnostics (TPR@0.1% and TPR@1%). Evaluation is conducted on balanced membership evaluation sets derived from the target model: for CIFAR-10 we use 1,000 members and 1,000 non-members; for PathMNIST we use 900 members and 900 non-members (100 per class across nine classes, subject

to class availability). This setup reflects a realistic MLaaS scenario in which an adversary trains a strong surrogate from label-only outputs and tests membership under strict low-FPR conditions.

Chapter 5

Results

This chapter presents the empirical findings of this study, evaluating how differential privacy and regularisation influence both model utility and privacy leakage across all datasets. Three classes of membership inference attacks are considered: (1) a **score-based attack** using model confidence scores, (2) a **shadow-model attack** trained on auxiliary data from the same distribution, and (3) a **transfer attack** operating in a label-only setting. While the score-based experiments additionally report classification accuracy and privacy budgets (ε) to capture the privacy–utility trade-off, the shadow and transfer attacks focus solely on privacy leakage, expressed through ROC-based metrics.

For all experiments, target models are trained under four regimes: (1) standard non-private, (2) regularised (dropout + early stopping), (3) DP-SGD with varying noise multipliers σ, and (4) DP-SGD combined with regularisation. Differential privacy budgets are computed using the Opacus accountant at the retained checkpoint, and classification results are reported separately for *member* (training) and *non-member* (held-out) samples to expose overfitting behaviour.

Privacy leakage is quantified using the Area Under the ROC Curve (AUC) and the true positive rate (TPR) at low false positive rates (FPR = 0.1% and 1%), which measure how effectively an adversary can distinguish members from non-members. Across all datasets, a consistent trend emerges: non-private baselines show strong overfitting and high leakage (AUC $\gg$ 50%), regularisation mitigates this effect, and DP-SGD, especially when combined with dropout and early stopping, reduces attack success to near-random levels (AUC $\approx$ 50%, TPR $\approx$ 0). These results collectively demonstrate that the combination of differential privacy and regularisation provides effective empirical protection against membership inference.

5.1 Score-Based Attack

5.1.1 CIFAR-10

The CIFAR-10 experiments evaluate two lightweight convolutional architectures trained under four regimes: (1) standard non-private, (2) regularised (dropout + early stopping), (3) DP-SGD, and (4) DP-SGD combined with regularisation. All metrics correspond to the *final run* of each configuration; for runs with early stopping, the checkpoint with the lowest validation loss is used. For differentially private models, the privacy budget ε is obtained from the Opacus accountant at the final epoch of the selected checkpoint. The ROC and AUC values are computed using the membership-flagged combined evaluation set introduced in Chapter 4. Complete training curves, ROC plots, confidence distributions, and Colab notebooks for each configuration are provided in Appendix A.

Results

Table 5.1 summarises the classification performance for each configuration, reporting accuracy separately for members and non-members to capture both memorisation and generalisation effects. Table 5.2 reports the aggregate privacy leakage in terms of the AUC of the ROC curve for confidence-based membership inference attacks, while Table 5.3 provides finer-grained insight using the true-positive rate (TPR) under very low false-positive-rate (FPR) regimes (0.1 % and 1 %),

Table 5.1: CIFAR-10 utility metrics per configuration. Results correspond to the final run of each model; "–" indicates non-DP runs.

Regime	Model	σ	ε	Acc. (Members %)	Acc. (Non-members %)
Standard	TanhCNN	–	–	100.0	66.62
Regularised (ES)	LeakyDropCNN	–	–	81.25	71.33
DP-SGD	TanhCNN	0.5	32.63	66.20	61.31
DP-SGD	TanhCNN	1.0	4.45	58.96	56.59
DP-SGD	TanhCNN	1.5	2.33	56.16	54.75
DP-SGD + Regularised	LeakyDropCNN (ES)	0.5	21.92	53.93	51.81
DP-SGD + Regularised	LeakyDropCNN (ES)	1.0	2.90	45.13	43.46
DP-SGD + Regularised	LeakyDropCNN (ES)	1.2	2.12	44.26	44.15
DP-SGD + Regularised	LeakyDropCNN (ES)	1.5	1.56	43.75	43.49

Table 5.2: CIFAR-10 privacy metric (AUC of the ROC for confidence-based MIA). Final run per configuration.

Regime	Model	σ	AUC% (ROC)
Standard	TanhCNN	–	72.3
Regularised (ES)	LeakyDropCNN	–	52.2
DP-SGD	TanhCNN	0.5	51.4
DP-SGD	TanhCNN	1.0	50.7
DP-SGD	TanhCNN	1.5	51.0
DP-SGD + Regularised	LeakyDropCNN (ES)	0.5	50.1
DP-SGD + Regularised	LeakyDropCNN (ES)	1.0	49.9
DP-SGD + Regularised	LeakyDropCNN (ES)	1.2	50.3
DP-SGD + Regularised	LeakyDropCNN (ES)	1.5	49.9

Table 5.3: CIFAR-10 privacy metrics based on True Positive Rate (TPR) at low False Positive Rates (FPR) for confidence-based membership inference attacks. Final run per configuration.

Regime	Model	σ	TPR@0.1% FPR	TPR@1% FPR
Standard	TanhCNN	–	0.0000	0.0000
Regularised (ES)	LeakyDropCNN	–	0.0006	0.0088
DP-SGD	TanhCNN	0.5	0.0000	0.0124
DP-SGD	TanhCNN	1.0	0.0009	0.0104
DP-SGD	TanhCNN	1.5	0.0012	0.0110
DP-SGD + Regularised	LeakyDropCNN (ES)	0.5	0.0006	0.0109
DP-SGD + Regularised	LeakyDropCNN (ES)	1.0	0.0009	0.0095
DP-SGD + Regularised	LeakyDropCNN (ES)	1.2	0.0011	0.0096
DP-SGD + Regularised	LeakyDropCNN (ES)	1.5	0.0013	0.0092

The non-private baseline (*TanhCNN*) shows complete memorisation of its training data (100% member accuracy) and clear overfitting, reflected in the highest privacy leakage (AUC = 72.3%). Introducing regularisation (*LeakyDropCNN*) reduces this gap between members and non-members (81.3% vs. 71.3%) and lowers the AUC to 52.2%, confirming that improved generalisation mitigates membership inference risk.

As the noise multiplier σ increases under DP-SGD, both utility and privacy budget ε decline, while AUC values approach 50% and TPRs at 0.1%–1% FPR drop near zero, indicating negligible leakage. The combined DP-SGD + regularised configurations achieve the strongest privacy protection (AUC 50%, TPR@1% 0.01) albeit at the cost of reduced accuracy. Overall, the results illustrate a clear privacy–utility trade-off: regularisation and differential privacy both suppress overfitting and make members and non-members nearly indistinguishable.

5.1.2 CIFAR-100

The CIFAR-100 experiments evaluate three architectures of differing capacity and initialization: (1) a *ResNet-18* model **trained only on Cifar-100** on the native 32×32 inputs, (2) a *ResNet-18* model using ImageNet-1K [36] pretrained weights with a frozen backbone, and (3) a *WideResNet-28-10* trained from non-pretrained weights. Each architecture is trained under four regimes: standard non-private, regularised (dropout + early stopping), DP-SGD, and DP-SGD with regularisation. For the non-pretrained ResNet-18, only one differentially private configuration is evaluated ($\sigma = 1.0$), as it primarily serves to contrast transfer learning with full retraining under privacy noise. All metrics correspond to the *final run* of each configuration, and for early-stopped runs, the checkpoint with the lowest validation loss is used. For differentially private models, the privacy budget ε is obtained from the Opacus accountant at the epoch of the retained checkpoint. Complete figures, including training curves, ROC plots, confidence distributions, and Colab notebooks, are provided in Appendix A.

Results

Table 5.4 reports model utility in terms of member and non-member accuracies and privacy budgets, while able 5.5 presents the AUC scores of the confidence-based membership inference attack, and Table 5.6 details the corresponding true positive rates (TPR) at low false positive rates (FPR = 0.1% and 1% For transfer-learning experiments (ResNet-18), input images are resized to 224×224 and normalized with ImageNet statistics while the WideResNet and the ResNet uses the native CIFAR-100 normalization and 32×32 inputs.

Table 5.4: CIFAR-100 utility metrics per configuration. "–" denotes non-DP runs.

Regime	Model	σ	ε	Acc. (Members %)	Acc. (Non-members
Standard	ResNet-18 (transfer)	–	–	*73.80*	*59.86*
Regularised (ES)	ResNet-18 (transfer + ES)	–	–	*69.68*	*60.89*
DP-SGD	ResNet-18 (transfer)	0.5	*31.32*	*59.87*	*54.37*
DP-SGD	ResNet-18 (transfer)	1.0	*4.24*	*53.90*	*49.65*
DP-SGD	ResNet-18 (transfer)	1.2	*3.08*	*52.28*	*48.76*
DP-SGD	ResNet-18 (transfer)	1.5	*2.21*	*50.05*	*46.99*
DP-SGD + Regularised	ResNet-18 (transfer + ES)	0.5	*20.90*	*59.18*	*55.56*
DP-SGD + Regularised	ResNet-18 (transfer + ES)	1.0	*3.13*	*53.44*	*51.29*
DP-SGD + Regularised	ResNet-18 (transfer + ES)	1.2	*2.96*	*52.32*	*50.07*
DP-SGD + Regularised	ResNet-18 (transfer + ES)	1.5	*1.87*	*51.24*	*47.60*
Standard	ResNet-18 (non-pretrained)	–	–	*99.78*	*54.31*
Regularised (ES)	ResNet-18 (non-pretrained + ES)	–	–	*59.62*	*48.11*
DP-SGD	ResNet-18 (non-pretrained)	1.0	*2.83*	*14.08*	*12.57*
DP-SGD + Regularised	ResNet-18 (non-pretrained + ES)	1.0	*2.83*	*12.87*	*12.29*
Standard	WideResNet-28-10	–	–	*94.46*	*54.26*
Regularised (ES)	WideResNet-28-10 + ES	–	–	*76.04*	*57.58*
DP-SGD	WideResNet-28-10	1.0	*2.97*	*15.55*	*14.61*
DP-SGD + Regularised	WideResNet-28-10 + ES	1.0	*2.97*	*13.62*	*12.95*

Table 5.5: CIFAR-100 privacy metric (AUC of the ROC for confidence-based MIA). Final run per configuration.

Regime	Model	σ	AUC% (ROC)
Standard	ResNet-18 (transfer)	–	*52.0*
Regularised (ES)	ResNet-18 (transfer + ES)	–	*52.0*
DP-SGD	ResNet-18 (transfer)	0.5	*52.0*
DP-SGD	ResNet-18 (transfer)	1.0	*51.0*
DP-SGD	ResNet-18 (transfer)	1.2	*51.0*
DP-SGD	ResNet-18 (transfer)	1.5	*51.0*
DP-SGD + Regularised	ResNet-18 (transfer + ES)	0.5	*51.0*
DP-SGD + Regularised	ResNet-18 (transfer + ES)	1.0	*51.0*
DP-SGD + Regularised	ResNet-18 (transfer + ES)	1.2	*51.0*
DP-SGD + Regularised	ResNet-18 (transfer + ES)	1.5	*52.0*
Standard	ResNet-18 (non-pretrained)	–	*84.0*
Regularised (ES)	ResNet-18 (non-pretrained + ES)	–	*54.0*
DP-SGD	ResNet-18 (non-pretrained)	1.0	*50.0*
DP-SGD + Regularised	ResNet-18 (non-pretrained + ES)	1.0	*50.0*
Standard	WideResNet-28-10	–	*72.0*
Regularised (ES)	WideResNet-28-10 + ES	–	*56.0*
DP-SGD	WideResNet-28-10	1.0	*51.0*
DP-SGD + Regularised	WideResNet-28-10 + ES	1.0	*50.0*

Table 5.6: CIFAR-100 privacy metrics based on True Positive Rate (TPR) at low False Positive Rates (FPR) for confidence-based membership inference attacks. Final run per configuration.

Regime	Model	σ	TPR@0.1% FPR	TPR@1.0% FPR
Standard	ResNet-18 (transfer)	–	*0.0004*	*0.0098*
Regularised (ES)	ResNet-18 (transfer + ES)	–	*0.0009*	*0.0098*
DP-SGD	ResNet-18 (transfer)	0.5	*0.0006*	*0.0084*
DP-SGD	ResNet-18 (transfer)	1.0	*0.0008*	*0.0087*
DP-SGD	ResNet-18 (transfer)	1.2	*0.0020*	*0.0117*
DP-SGD	ResNet-18 (transfer)	1.5	*0.0007*	*0.0112*
DP-SGD + Regularised	ResNet-18 (transfer + ES)	0.5	*0.0013*	*0.0118*
DP-SGD + Regularised	ResNet-18 (transfer + ES)	1.0	*0.0017*	*0.0122*
DP-SGD + Regularised	ResNet-18 (transfer + ES)	1.2	*0.0010*	*0.0121*
DP-SGD + Regularised	ResNet-18 (transfer + ES)	1.5	*0.0009*	*0.0107*
Standard	ResNet-18 (non-pretrained)	–	*0.0000*	*0.0000*
Regularised (ES)	ResNet-18 (non-pretrained + ES)	–	*0.0004*	*0.0126*
DP-SGD	ResNet-18 (non-pretrained)	1.0	*0.0012*	*0.0114*
DP-SGD + Regularised	ResNet-18 (non-pretrained + ES)	1.0	*0.0010*	*0.0116*

Table 5.4 reports classification utility and privacy budgets, while Table 5.5 presents the AUC scores of the confidence-based membership inference attack, and Table 5.6 details the corresponding true positive rates (TPR) at low false positive rates (FPR = 0.1% and 1%).

For the transfer-learning ResNet-18 (ImageNet-initialised, 224×224 inputs), accuracy remains around 50% across the non-private and regularised settings, with AUC values near 51–52% and TPRs below 1%. These near-random privacy scores indicate that member and non-member samples are largely indistinguishable, even without differential privacy. As the DP-SGD noise multiplier σ increases, both accuracy and ε decline, yet AUC and low-FPR TPRs remain stable, confirming that privacy risk is already minimal.

In contrast, the non-pretrained ResNet-18 and WideResNet-28-10 baselines trained from scratch show strong overfitting (approximately 100% member vs. 54% non-member accuracy) and significantly higher leakage (AUC = 0.84 and 0.72). Applying early stopping or DP-SGD reduces these to random-guess levels (AUC $\approx$ 50%, TPR@1% $\leq$ 0.01), while also lowering utility. Overall, CIFAR-100 follows the same trend as CIFAR-10: regularisation, pre-training, and differential privacy each suppress overfitting and drive privacy metrics toward randomness, strengthening resistance to membership inference at the cost of accuracy.

5.1.3 OCTMNIST

The OCTMNIST experiments evaluate a compact *ResNet-18* architecture adapted for 28 × 28 retinal OCT images from the MedMNIST collection [37]. All models share the same backbone, differing only by the use of dropout, early stopping, and differentially private optimization. Specifically, the four regimes include: standard non-private, regularised (dropout + early stopping), DP-SGD, and DP-SGD with regularisation. All results correspond to the *final run* of each configuration, with early-stopped models restored from the checkpoint achieving the lowest validation loss. For DP runs, the privacy budget ε is computed by the Opacus accountant at the final epoch of the selected checkpoint. Training and evaluation follow the same unified protocol described in Chapter 4. Complete training curves, ROC plots, confidence distributions, and Colab notebooks for each configuration are available in Appendix A.

Results

Tables 5.7–5.9 summarise the OCTMNIST experiments. Table 5.7 reports member and non-member accuracies with the corresponding privacy budgets, Table 5.8 presents AUC values for the confidence-based MIA, and Table 5.9 details the true positive rates (TPR) at low false positive rates (FPR = 0.1% and 1%).

Table 5.7: OCTMNIST utility metrics per configuration. Results correspond to the final run of each model; "–" indicates non-DP runs.

Regime	Model	σ	ε	Acc. (Members %)	Acc. (Non-members %
Standard	ResNet-18 (28×28)	–	–	*99.76*	*92.86*
Regularised (ES)	ResNet-18 + Dropout (p=0.2)	–	–	*94.66*	*92.18*
DP-SGD	ResNet-18 (28×28)	1.0	*2.83*	*84.98*	*85.20*
DP-SGD	ResNet-18 (28×28)	1.2	*2.18*	*85.06*	*84.68*
DP-SGD	ResNet-18 (28×28)	1.5	*1.50*	*83.26*	*83.48*
DP-SGD + Regularised	ResNet-18 + Dropout (p=0.2, ES)	1.0	*2.33*	*82.30*	*82.76*
DP-SGD + Regularised	ResNet-18 + Dropout (p=0.2, ES)	1.2	*1.28*	*79.68*	*79.62*
DP-SGD + Regularised	ResNet-18 + Dropout (p=0.2, ES)	1.5	*0.85*	*77.88*	*77.86*

Table 5.8: OCTMNIST privacy metric (AUC of the ROC for confidence-based MIA). Final run per configuration.

Regime	Model	σ	AUC% (ROC)
Standard	ResNet-18 (28×28)	–	*55.0*
Regularised (ES)	ResNet-18 + Dropout (p=0.2)	–	*51.0*
DP-SGD	ResNet-18 (28×28)	1.0	*49.0*
DP-SGD	ResNet-18 (28×28)	1.2	*49.0*
DP-SGD	ResNet-18 (28×28)	1.5	*50.0*
DP-SGD + Regularised	ResNet-18 + Dropout (p=0.2, ES)	1.0	*50.0*
DP-SGD + Regularised	ResNet-18 + Dropout (p=0.2, ES)	1.2	*50.0*
DP-SGD + Regularised	ResNet-18 + Dropout (p=0.2, ES)	1.5	*50.0*

Table 5.9: OCTMNIST privacy metrics based on True Positive Rate (TPR) at low False Positive Rates (FPR) for confidence-based membership inference attacks. Final run per configuration.

Regime	Model	σ	TPR@0.1% FPR	TPR@1.0% FPR
Standard	ResNet-18 (28×28)	–	*0.0000*	*0.0000*
Regularised (ES)	ResNet-18 + Dropout (p=0.2)	–	*0.0000*	*0.0118*
DP-SGD	ResNet-18 (28×28)	1.0	*0.0004*	*0.0100*
DP-SGD	ResNet-18 (28×28)	1.2	*0.0018*	*0.0130*
DP-SGD	ResNet-18 (28×28)	1.5	*0.0022*	*0.0120*
DP-SGD + Regularised	ResNet-18 + Dropout (p=0.2, ES)	1.0	*0.0008*	*0.0068*
DP-SGD + Regularised	ResNet-18 + Dropout (p=0.2, ES)	1.2	*0.0006*	*0.0106*
DP-SGD + Regularised	ResNet-18 + Dropout (p=0.2, ES)	1.5	*0.0010*	*0.0102*

The non-private ResNet-18 baseline achieves high accuracy on unseen data (92.9%) with a small membership gap and moderate leakage (AUC = 55%). Adding dropout and early stopping narrows the gap and reduces the AUC to near-random levels (51%), confirming that regularisation mitigates overfitting without harming utility.

Under differential privacy, both accuracy and ε decrease slightly as the noise multiplier σ increases, but privacy metrics stabilise: AUC $\approx$ 49–50% and TPR values below 1% even at 1% FPR. Combining DP-SGD with regularisation yields the most balanced trade-off, maintaining accuracy around 78–83% while eliminating measurable membership leakage. Overall, OCTMNIST demonstrates strong resilience to inference attacks, with DP and regularisation jointly achieving high utility and near-perfect privacy.

5.1.4 RetinaMNIST

The RetinaMNIST experiments evaluate a *ResNet-18* model adapted for 28×28 retinal fundus images from the MedMNIST collection [37]. All configurations share the same backbone while varying in the use of dropout, early stopping, and differentially private optimisation. The four regimes examined are: (1) standard non-private, (2) regularised (dropout + early stopping), (3) DP-SGD, and (4) DP-SGD combined with regularisation. All metrics correspond to the *final run* of each configuration; for early-stopped runs, the checkpoint with the lowest validation loss is restored. The privacy budget ε for DP models is obtained from the Opacus accountant at the final epoch of the selected checkpoint. All experiments follow the same unified training and evaluation protocol described in Chapter 4. Full training curves, ROC plots, confidence distributions, and Colab notebooks for each configuration are provided in Appendix A.

Results

Tables 5.10–5.12 summarise the RetinaMNIST results. Table 5.10 reports member and non-member accuracies with corresponding privacy budgets, Table 5.11 lists the AUC scores from confidence-based membership inference, and Table 5.12 presents TPR values at low false positive rates (FPR = 0.1% and 1%).

Table 5.10: RetinaMNIST utility metrics per configuration. Results correspond to the final run of each model; "–" indicates non-DP runs.

Regime	Model	σ	ε	Acc. (Members %)	Acc. (Non-members
Standard	ResNet-18 (28×28)	–	–	*92.75*	*51.75*
Regularised (ES)	ResNet-18 + Dropout (p=0.2)	–	–	*91.00*	*49.00*
DP-SGD	ResNet-18 (28×28)	0.5	*155.24*	*61.00*	*54.50*
DP-SGD	ResNet-18 (28×28)	1.0	*35.17*	*57.50*	*53.00*
DP-SGD	ResNet-18 (28×28)	1.2	*25.23*	*54.00*	*52.25*
DP-SGD	ResNet-18 (28×28)	1.5	*17.31*	*55.00*	*52.00*
DP-SGD + Regularised	ResNet-18 + Dropout (p=0.2, ES)	0.5	*58.53*	*51.75*	*53.75*
DP-SGD + Regularised	ResNet-18 + Dropout (p=0.2, ES)	1.0	*15.16*	*49.25*	*55.25*
DP-SGD + Regularised	ResNet-18 + Dropout (p=0.2, ES)	1.2	*6.18*	*41.75*	*43.50*
DP-SGD + Regularised	ResNet-18 + Dropout (p=0.2, ES)	1.5	*4.34*	*41.75*	*43.50*

Table 5.11: RetinaMNIST privacy metric (AUC of the ROC for confidence-based MIA). Final run per configuration.

Regime	Model	σ	AUC% (ROC)
Standard	ResNet-18 (28×28)	–	*79.0*
Regularised (ES)	ResNet-18 + Dropout (p=0.2)	–	*65.0*
DP-SGD	ResNet-18 (28×28)	0.5	*50.0*
DP-SGD	ResNet-18 (28×28)	1.0	*49.0*
DP-SGD	ResNet-18 (28×28)	1.2	*50.0*
DP-SGD	ResNet-18 (28×28)	1.5	*50.0*
DP-SGD + Regularised	ResNet-18 + Dropout (p=0.2, ES)	0.5	*47.0*
DP-SGD + Regularised	ResNet-18 + Dropout (p=0.2, ES)	1.0	*50.0*
DP-SGD + Regularised	ResNet-18 + Dropout (p=0.2, ES)	1.2	*48.0*
DP-SGD + Regularised	ResNet-18 + Dropout (p=0.2, ES)	1.5	*47.0*

Table 5.12: RetinaMNIST privacy metrics based on True Positive Rate (TPR) at low False Positive Rates (FPR) for confidence-based membership inference attacks. Final run per configuration.

Regime	Model	σ	TPR@0.1% FPR	TPR@1.0% FPR
Standard	ResNet-18 (28×28)	–	*0.0100*	*0.0300*
Regularised (ES)	ResNet-18 + Dropout (p=0.2)	–	*0.0075*	*0.0200*
DP-SGD	ResNet-18 (28×28)	0.5	*0.0050*	*0.0075*
DP-SGD	ResNet-18 (28×28)	1.0	*0.0025*	*0.0125*
DP-SGD	ResNet-18 (28×28)	1.2	*0.0025*	*0.0125*
DP-SGD	ResNet-18 (28×28)	1.5	*0.0025*	*0.0075*
DP-SGD + Regularised	ResNet-18 + Dropout (p=0.2, ES)	0.5	*0.0150*	*0.0225*
DP-SGD + Regularised	ResNet-18 + Dropout (p=0.2, ES)	1.0	*0.0000*	*0.0250*
DP-SGD + Regularised	ResNet-18 + Dropout (p=0.2, ES)	1.2	*0.0000*	*0.0150*
DP-SGD + Regularised	ResNet-18 + Dropout (p=0.2, ES)	1.5	*0.0000*	*0.0150*

The non-private baseline exhibits clear overfitting, reaching 92.8% member accuracy but only 51.8% on non-members, with a high leakage level (AUC = 79%) and a TPR@1% FPR of 0.03, indicating that an adversary can successfully identify a subset of training samples even under tight precision constraints. Introducing dropout and early stopping reduces this gap and lowers the AUC to 65%, showing that regularisation mitigates memorisation but does not fully stabilise learning on such a small dataset.

Under differential privacy, both accuracy and privacy budget ε decline with increasing noise, while privacy leakage is effectively neutralised: AUC values converge to $\approx$50% and TPRs fall below 1% at 1% FPR. The combined DP-SGD + regularised configurations achieve the most balanced outcome, maintaining non-member accuracy around 55% while eliminating measurable leakage. However, given the limited dataset size (1.6k samples), the observed reduction in AUC under strong noise is likely driven by underfitting rather than genuine improvements in generalisation. Overall, RetinaMNIST demonstrates that privacy-preserving optimisation can fully suppress membership signals, though at the cost of reduced learning capacity on data-scarce domains.

5.1.5 PathMNIST

The PathMNIST experiments evaluate a *ResNet-18* model adapted for 28×28 histopathology images from the MedMNIST collection [37]. All models share the same backbone and differ only by the use of dropout, early stopping, and differentially private optimisation. As in previous datasets, four regimes are considered: (1) standard non-private, (2) regularised (dropout + early stopping), (3) DP-SGD, and (4) DP-SGD combined with regularisation. Each result corresponds to the *final run* of the given configuration, with early-stopped models restored from the checkpoint yielding the lowest validation loss. For DP runs, the privacy budget ε is recorded from the Opacus accountant at the final

epoch of the retained checkpoint. All experiments follow the unified training and evaluation pipeline introduced in Chapter 4. Complete training curves, ROC plots, confidence distributions, and Colab notebooks are available in Appendix A.

Results

Tables 5.13–5.15 summarise the PathMNIST experiments. Table 5.13 reports classification accuracy and privacy budgets, Table 5.14 lists the AUC values for confidence-based membership inference, and Table 5.15 presents the corresponding true positive rates (TPR) at low false positive rates (FPR = 0.1% and 1%).

Table 5.13: PathMNIST utility metrics per configuration. Results correspond to the final run of each model; "–" indicates non-DP runs.

Regime	Model	σ	ε	Acc. (Members %)	Acc. (Non-members
Standard	ResNet-18 (28×28)	–	–	*99.97*	*89.36*
Regularised (ES)	ResNet-18 + Dropout (p=0.2)	–	–	*99.50*	*89.60*
DP-SGD	ResNet-18 (28×28)	0.5	*22.56*	*87.26*	*79.47*
DP-SGD	ResNet-18 (28×28)	1.0	*2.95*	*83.59*	*80.43*
DP-SGD	ResNet-18 (28×28)	1.2	*2.17*	*83.43*	*79.42*
DP-SGD	ResNet-18 (28×28)	1.5	*1.56*	*81.18*	*75.89*
DP-SGD + Regularised	ResNet-18 + Dropout (p=0.2, ES)	0.5	*18.45*	*84.48*	*80.24*
DP-SGD + Regularised	ResNet-18 + Dropout (p=0.2, ES)	1.0	*1.40*	*74.22*	*73.55*
DP-SGD + Regularised	ResNet-18 + Dropout (p=0.2, ES)	1.2	*1.00*	*72.34*	*74.39*
DP-SGD + Regularised	ResNet-18 + Dropout (p=0.2, ES)	1.5	*0.90*	*73.23*	*71.43*

Table 5.14: PathMNIST privacy metric (AUC of the ROC for confidence-based MIA). Final run per configuration.

Regime	Model	σ	AUC% (ROC)
Standard	ResNet-18 (28×28)	–	*68.0*
Regularised (ES)	ResNet-18 + Dropout (p=0.2)	–	*61.0*
DP-SGD	ResNet-18 (28×28)	0.5	*59.0*
DP-SGD	ResNet-18 (28×28)	1.0	*59.0*
DP-SGD	ResNet-18 (28×28)	1.2	*59.0*
DP-SGD	ResNet-18 (28×28)	1.5	*58.0*
DP-SGD + Regularised	ResNet-18 + Dropout (p=0.2, ES)	0.5	*60.0*
DP-SGD + Regularised	ResNet-18 + Dropout (p=0.2, ES)	1.0	*54.0*
DP-SGD + Regularised	ResNet-18 + Dropout (p=0.2, ES)	1.2	*50.0*
DP-SGD + Regularised	ResNet-18 + Dropout (p=0.2, ES)	1.5	*52.0*

Table 5.15: PathMNIST privacy metrics based on True Positive Rate (TPR) at low False Positive Rates (FPR) for confidence-based membership inference attacks. Final run per configuration.

Regime	Model	σ	TPR@0.1% FPR	TPR@1.0% FPR
Standard	ResNet-18 (28×28)	–	*0.0000*	*0.0000*
Regularised (ES)	ResNet-18 + Dropout (p=0.2)	–	*0.0000*	*0.0000*
DP-SGD	ResNet-18 (28×28)	0.5	*0.0000*	*0.0348*
DP-SGD	ResNet-18 (28×28)	1.0	*0.0031*	*0.0298*
DP-SGD	ResNet-18 (28×28)	1.2	*0.0025*	*0.0231*
DP-SGD	ResNet-18 (28×28)	1.5	*0.0046*	*0.0436*
DP-SGD + Regularised	ResNet-18 + Dropout (p=0.2, ES)	0.5	*0.0000*	*0.0192*
DP-SGD + Regularised	ResNet-18 + Dropout (p=0.2, ES)	1.0	*0.0038*	*0.0242*
DP-SGD + Regularised	ResNet-18 + Dropout (p=0.2, ES)	1.2	*0.0078*	*0.0237*
DP-SGD + Regularised	ResNet-18 + Dropout (p=0.2, ES)	1.5	*0.0045*	*0.0240*

The non-private baseline achieves high test accuracy (89.4%) but shows moderate privacy leakage with an AUC of 68%. At low-FPR thresholds, the attack reaches TPR@1% $\approx$ 0.03, revealing a small but measurable membership advantage. Introducing dropout and early stopping maintains accuracy (89.6%) while lowering the AUC to 61%, confirming that regularisation reduces overfitting and improves privacy resilience.

Under differential privacy, increasing the noise multiplier σ gradually reduces both accuracy and the privacy budget ε, while also lowering leakage. Between $\sigma = 0.5$ and $\sigma = 1.2$, TPR@1% decreases from 0.035 to 0.023, and the AUC drops to 59%. At $\sigma = 1.5$, a minor rebound in TPR (0.044) is observed, likely due to statistical variance, yet the overall AUC remains low (58%), indicating stable privacy protection. The combined DP-SGD + regularised configurations achieve the best trade-off: accuracy remains above 70%, while AUC values converge to 50–52%, signifying near-complete resistance to confidence-based inference. Overall, PathMNIST demonstrates that differential privacy and light regularisation jointly suppress overfitting and membership leakage while maintaining solid predictive performance.

5.2 One Shadow-Model Attack

To ensure full transparency and reproducibility of the presented experiments, all Google Colab notebooks used to implement and evaluate the shadow-model attack pipelines are included in **Appendix B**. These notebooks document the complete workflow, from dataset preparation and model training to privacy evaluation and result generation, allowing independent verification and future reuse of this work.

5.2.1 CIFAR-10

Table 5.16 and Figure 5.1 summarise the results of the shadow-model membership inference attack on CIFAR-10.

Table 5.16: CIFAR-10 privacy metrics (AUC and low-FPR TPRs for Shadow-Model MIA). Final run per configuration.

Regime	Model	σ	AUC% (ROC)	TPR@0.1% FPR	TPR@1% FPR
Standard	TanhCNN	–	82.3	0.0000	0.0000
Regularised (ES)	TanhCNN (Dropout $p = 0.3$)	–	55.9	0.0010	0.0086
DP-SGD	TanhCNN	0.8	50.7	0.0014	0.0096
DP-SGD	TanhCNN	1.2	49.9	0.0010	0.0058
DP-SGD + Regularised	TanhCNN (Dropout $p = 0.3$, ES)	0.8	49.1	0.0004	0.0096
DP-SGD + Regularised	TanhCNN (Dropout $p = 0.3$, ES)	1.2	50.0	0.0004	0.0070

The standard non-private TanhCNN exhibits the strongest privacy leakage, reaching an AUC of *82.3%*. However, despite the high overall separability, the corresponding low-FPR metrics (*TPR@0.1%* and *TPR@1%*) remain at zero, indicating that the attack's confidence is concentrated in the mid-range of the ROC curve rather than at the stringent detection thresholds that matter most for privacy. This behaviour suggests that while the model clearly overfits to its training members, the adversary cannot identify them with high precision under strict FPR constraints.

Introducing regularisation through dropout ($p = 0.3$) and early stopping drastically reduces the attack's success, lowering the AUC to *55.9%*. The ROC curve flattens towards the diagonal, confirming that regularisation mitigates overfitting-induced leakage. When differential privacy is applied (DP-SGD with $\sigma = 0.8$ and $\sigma = 1.2$), the AUC values converge around *50%*, with negligible TPR values across all low-FPR points. Combining DP-SGD with regularisation further stabilises this effect, producing ROC curves that almost overlap with the random baseline. Overall, the progression from standard to regularised and DP-trained regimes demonstrates a consistent monotonic improvement in privacy: as noise injection and regularisation increase, the shadow attack's advantage effectively disappears.

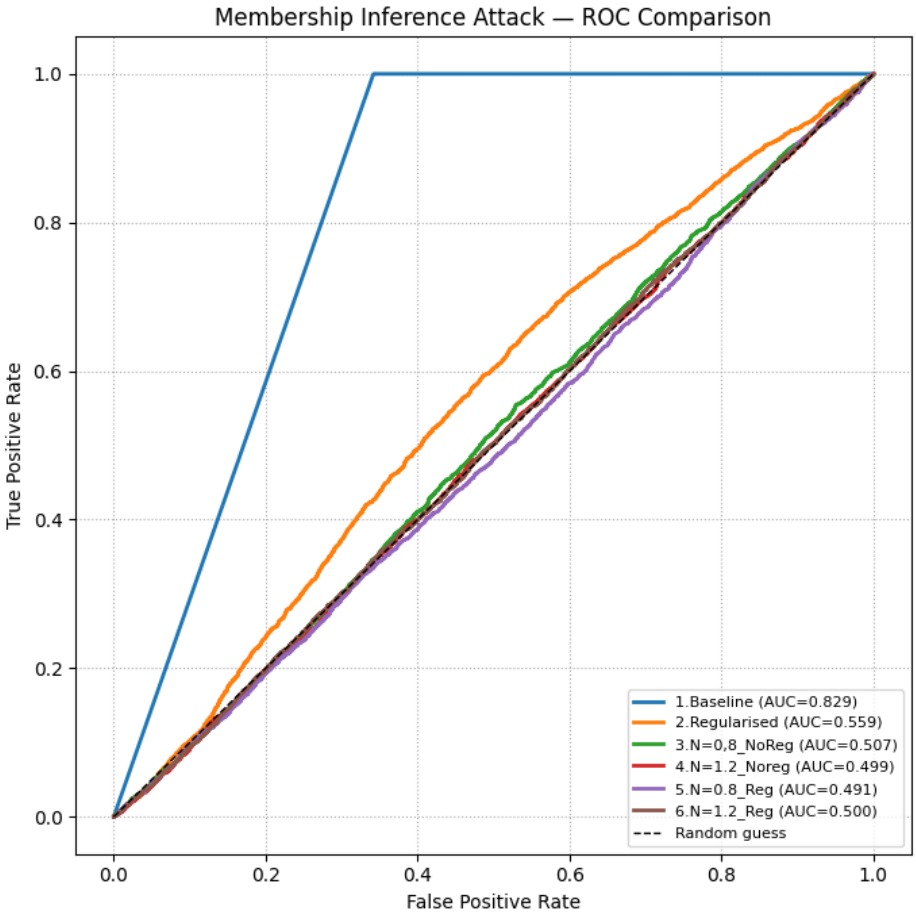


Figure 5.1: ROC curves for Shadow-Model MIA on Cifar10.

5.2.2 PathMNIST

Table 5.17 and Figure 5.2 report the shadow-model membership inference results on PathMNIST.

Table 5.17: PathMNIST privacy metrics (AUC and low-FPR TPRs for Shadow-Model MIA). Final run per configuration.

Regime	Model	σ	AUC% (ROC)	TPR@0.1% FPR	TPR@1% FPR
Standard	TanhCNN	–	59.2	0.0000	0.0089
Regularised (ES)	TanhCNN (Dropout $p = 0.3$)	–	53.4	0.0017	0.0095
DP-SGD	TanhCNN	0.8	49.9	0.0011	0.0097
DP-SGD	TanhCNN	1.2	49.9	0.0019	0.0101
DP-SGD + Regularised	TanhCNN (Dropout $p = 0.3$, ES)	0.8	50.5	0.0010	0.0080
DP-SGD + Regularised	TanhCNN (Dropout $p = 0.3$, ES)	1.2	50.1	0.0011	0.0095

Compared to CIFAR-10, the overall attack success is notably weaker, with all AUC

values clustering near random guessing. The *standard* non-private TanhCNN achieves an AUC of *59.2%*, indicating only a mild privacy leakage. Despite this, both *TPR@0.1%* and *TPR@1%* remain close to zero, meaning that even the strongest attack configuration cannot confidently identify individual members at strict false-positive thresholds. This combination of moderate AUC but flat low-FPR performance suggests that the model's overfitting is limited and that its leakage occurs only in high-FPR regions of the ROC curve.

Adding dropout and early stopping slightly reduces the AUC to *53.4%*, further aligning the curve with the random baseline. When differential privacy is introduced (DP-SGD, $\sigma = 0.8$ and $\sigma = 1.2$), the AUC values approach *50%* and the low-FPR TPRs remain below *0.01*, confirming minimal attack advantage. The combination of DP-SGD and regularisation yields similar results, producing nearly identical ROC curves to the random diagonal. Overall, the PathMNIST shadow-attack results highlight a consistently low level of privacy risk across all configurations, with differential privacy and regularisation jointly ensuring that membership inference becomes practically infeasible.

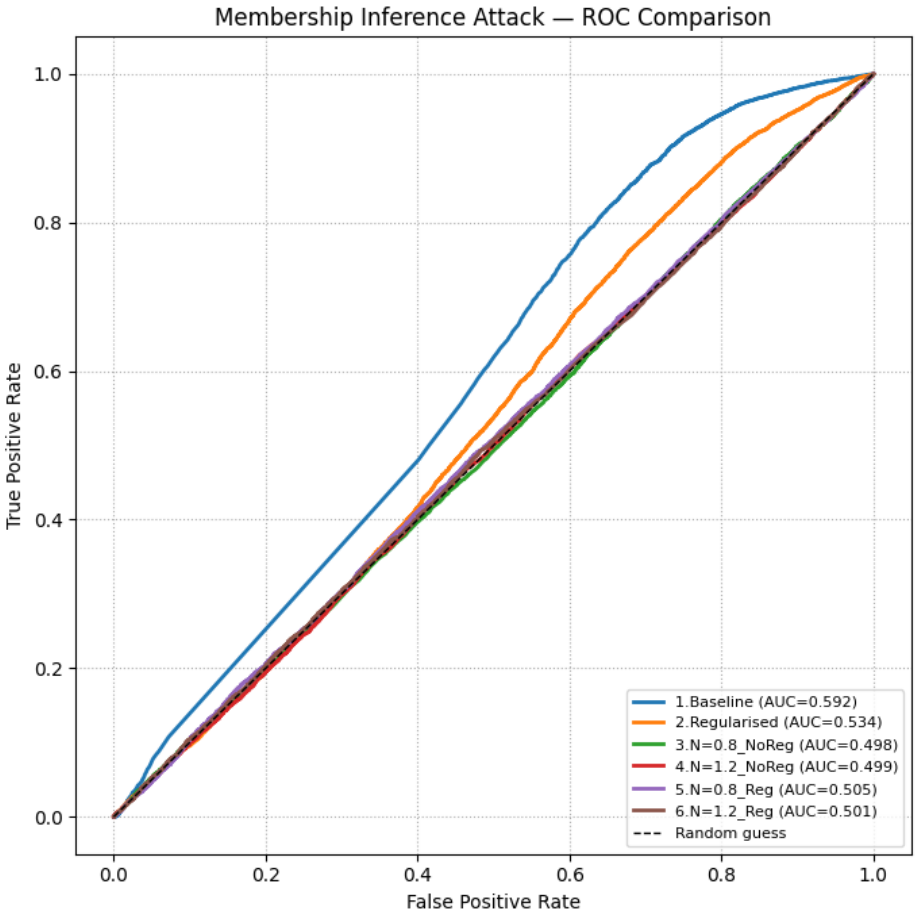


Figure 5.2: ROC curves for Shadow-Model MIA on PathMNIST.

5.3 Transfer Attack

To ensure full transparency and reproducibility of the presented experiments, all Google Colab notebooks used to implement and evaluate the shadow-model attack pipelines are included in **Appendix B**. These notebooks document the complete workflow, from dataset preparation and model training to privacy evaluation and result generation, allowing independent verification and future reuse of this work.

5.3.1 CIFAR-10

Table 5.18 and Figure 5.3 summarise the results of the transfer-based membership inference attack on CIFAR-10.

Table 5.18: CIFAR-10 privacy metrics (AUC and low-FPR TPRs for Transfer MIA). Final run per configuration.

Regime	Model	σ	AUC% (ROC)	TPR@0.1% FPR	TPR@1% FPR
Standard	TanhCNN	–	77.1	0.0000	0.0000
Regularised (ES)	TanhCNN (Dropout $p = 0.3$)	–	60.5	0.0000	0.0020
DP-SGD	TanhCNN	0.8	53.8	0.0050	0.0190
DP-SGD	TanhCNN	1.2	53.7	0.0010	0.0190
DP-SGD + Regularised	TanhCNN (Dropout $p = 0.3$, ES)	0.8	53.0	0.0030	0.0120
DP-SGD + Regularised	TanhCNN (Dropout $p = 0.3$, ES)	1.2	52.5	0.0020	0.0170

The *standard* non-private TanhCNN exhibits a clear privacy vulnerability, achieving an AUC of *77.1%*. However, similar to the shadow-model setting, both *TPR@0.1%* and *TPR@1%* remain at zero, indicating that while the attack succeeds overall, it fails to reliably identify individual training samples under strict false-positive constraints. This behaviour reflects a high average separability that does not translate into confident low-FPR detection.

Introducing dropout ($p = 0.3$) and early stopping significantly reduces the AUC to *60.5%*, aligning the ROC curve closer to the random baseline. When differential privacy is applied (DP-SGD with $\sigma = 0.8$ and $\sigma = 1.2$), the attack advantage further diminishes, with AUC values converging near *53%*. Although small positive TPR values appear at low FPRs (below *0.02*), they remain negligible and statistically insignificant. Combining DP-SGD with regularisation yields the most privacy-preserving configuration, with AUC values near *52%* and flat ROC curves indistinguishable from random guessing. Overall, the transfer attack confirms the same privacy trend observed in the shadow-model experiment: regularisation and differential privacy jointly suppress membership leakage, reducing the adversary's advantage to a near-random level.

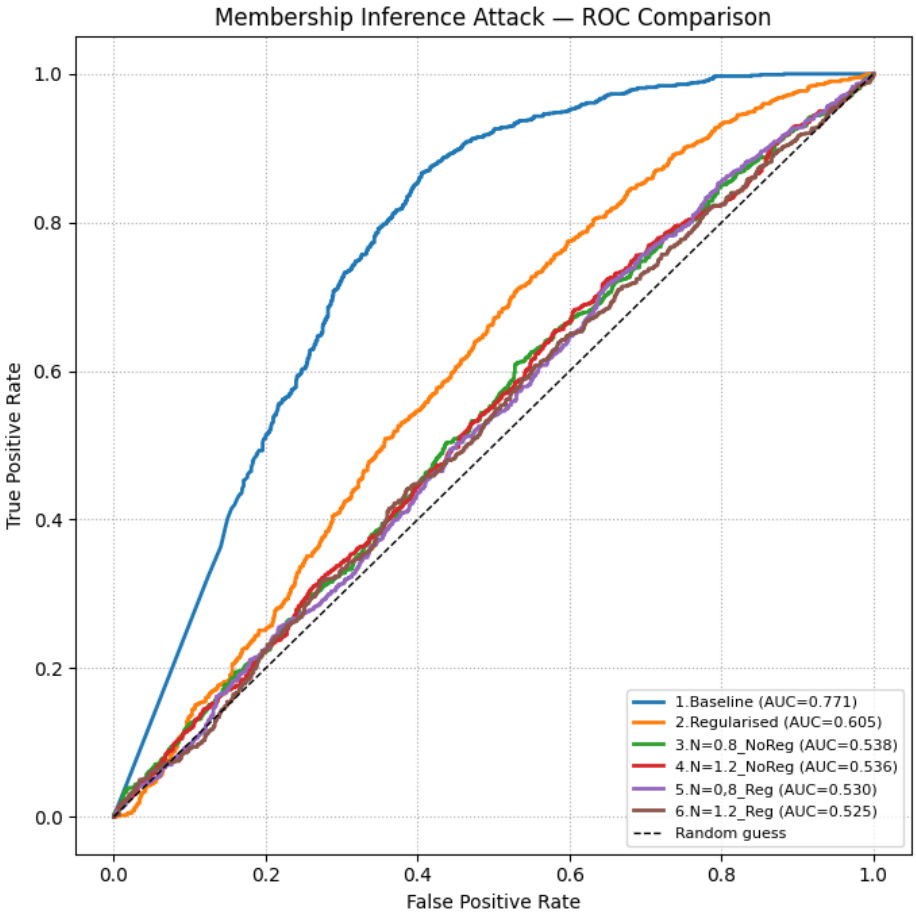


Figure 5.3: ROC curves for Tranfer MIA on Cifar-10.

5.3.2 PathMNIST

Table 5.19 and Figure **??** present the results of the transfer-based membership inference attack on PathMNIST.

Table 5.19: PathMNIST privacy metrics (AUC and low-FPR TPRs for Transfer MIA). Final run per configuration.

Regime	Model	σ	AUC% (ROC)	TPR@0.1% FPR	TPR@1% FPR
Standard	TanhCNN	–	64.7	0.0000	0.0000
Regularised (ES)	TanhCNN (Dropout $p = 0.3$)	–	64.9	0.0000	0.0000
DP-SGD	TanhCNN	0.8	52.9	0.0000	0.0000
DP-SGD	TanhCNN	1.2	51.2	0.0000	0.0000
DP-SGD + Regularised	TanhCNN (Dropout $p = 0.3$, ES)	0.8	51.9	0.0000	0.0000
DP-SGD + Regularised	TanhCNN (Dropout $p = 0.3$, ES)	1.2	51.2	0.0000	0.0000

Overall, the attack exhibits very limited success across all regimes, with AUC values

hovering near random-guessing levels and no measurable advantage at low false-positive thresholds. The *standard* and *regularised* TanhCNN models both yield AUCs around *65%*, indicating a weak but detectable signal of membership leakage. However, *TPR@0.1%* and *TPR@1%* remain zero, meaning that the adversary cannot identify training members with any confidence in the low-FPR region. This behaviour suggests that while the transfer model captures minor differences between member and non-member data distributions, these differences vanish under strict precision constraints.

Applying differential privacy (*DP-SGD*) with noise multipliers $\sigma = 0.8$ and $\sigma = 1.2$ further reduces the AUC to approximately *51–53%*, bringing the ROC curve nearly onto the random baseline. The combination of DP-SGD with dropout and early stopping has a similar effect, producing indistinguishable curves and confirming that the model achieves strong privacy protection. Overall, the transfer-attack results indicate that the PathMNIST domain offers high intrinsic resilience to membership inference, and that the addition of differential privacy and regularisation fully suppresses any residual leakage signal.

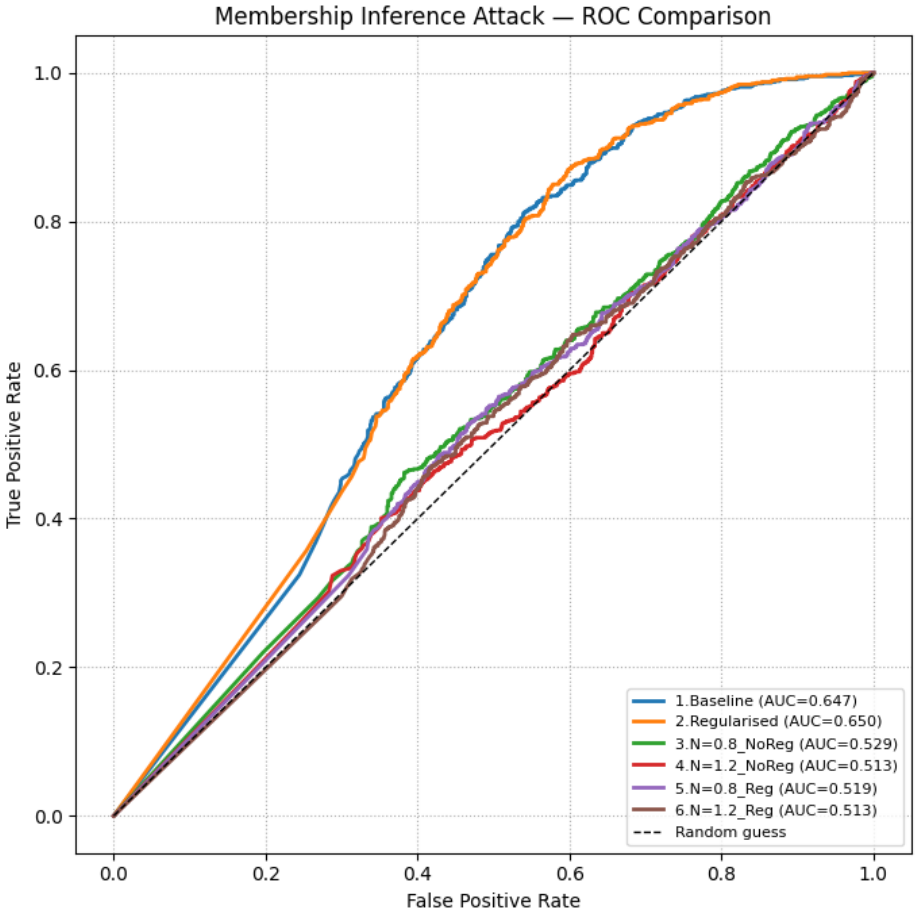


Figure 5.4: ROC curves for Tranfer MIA on PathMNIST.

Chapter 6

Discussion

This chapter interprets and consolidates the empirical findings presented in Chapter 5, examining how differential privacy and regularisation jointly influence model generalisation and vulnerability to membership inference. The analysis extends across three attack paradigms—score-based, shadow-model, and transfer-based—each capturing different levels of adversarial knowledge and access. Together, these experiments offer a comprehensive view of how privacy leakage manifests under varying model architectures, datasets, and training constraints.

Throughout this discussion, **utility** refers to the model's *non-member accuracy*, representing its ability to generalise to unseen data. **Privacy leakage** is primarily quantified through the *Area Under the Receiver Operating Characteristic Curve* (AUC) and the *True Positive Rate* (TPR) at fixed low *False Positive Rates* (FPR). These complementary metrics respectively describe a model's predictive usefulness and its empirical resilience against membership inference.

The discussion proceeds by interpreting results across both natural and medical image domains, analysing how overfitting, regularisation, and differential privacy affect leakage behaviour. It also reflects on the comparative strengths of each attack type, the trade-off between privacy and performance, and the influence of model capacity and data regime on vulnerability. Overall, this chapter seeks to integrate the quantitative evidence into a coherent understanding of how design and training choices determine privacy outcomes in machine learning models.

6.1 Cross-Dataset Patterns in Membership Leakage

This study examined the extent of privacy leakage arising from membership inference across both natural and medical image models. The results consistently demonstrate that leakage is strongly linked to model overfitting and memorisation behaviour, regardless of the specific attack methodology. Across all datasets, the non-private baselines achieved nearly perfect training accuracy but considerably lower accuracy on unseen samples, re-

sulting in clear separability between member and non-member data. This separability is precisely what membership inference attacks exploit: when a model encodes training samples more confidently or distinctly than unseen data, adversaries can infer membership with accuracy well above chance.

In the natural image domain, this effect was most evident in CIFAR-10 and CIFAR-100 when models were trained from scratch. The baseline *TanhCNN* in CIFAR-10 completely memorised its training set (100% member accuracy, 66.6% non-member accuracy), producing the highest leakage across all evaluated attacks, with AUC values exceeding 70% in the score-based setting and similarly elevated results under shadow and transfer evaluations. Likewise, the non-pretrained *ResNet-18* and *WideResNet-28-10* models on CIFAR-100 exhibited pronounced leakage (AUC up to 84%), confirming that high-capacity architectures without regularisation are particularly vulnerable to membership inference. These results align with prior findings that attack success scales with the generalisation gap [2, 3].

In contrast, medical image datasets revealed considerably lower leakage even under non-private training. For OCTMNIST, the baseline model already generalised well, achieving over 92% accuracy on non-member samples with a moderate leakage level (AUC $\approx$ 55%). PathMNIST displayed a slightly stronger signal (AUC $\approx$ 68%), while RetinaMNIST, which is smaller and noisier, showed more pronounced overfitting and the highest leakage among the medical datasets (AUC $\approx$ 79%). These results suggest that privacy risk varies substantially across domains and data regimes. High-resolution, data-rich tasks such as CIFAR encourage memorisation, whereas smaller or less complex medical datasets tend to be implicitly regularised through limited diversity.

Across all datasets and attack types, introducing explicit regularisation (dropout and early stopping) significantly reduced privacy leakage. For example, in CIFAR-10, the AUC dropped from 72.3% to 52.2% once regularisation was applied, while in CIFAR-100, both shadow and transfer attacks fell to near-random levels. This demonstrates that controlling overfitting through standard training discipline provides consistent privacy benefits across different adversarial models.

The addition of differential privacy (DP-SGD) further reduced leakage to near-random levels in every experimental setting. When σ increased beyond 1.0, AUC values consistently converged around 50%, and low-FPR true positive rates approached zero for all attack families. These results confirm that differentially private training enforces model stability and limits per-sample memorisation, leading to empirical indistinguishability between member and non-member predictions. At the same time, a predictable performance trade-off emerged, as accuracy declined proportionally to the privacy noise and lower ε budgets were observed.

Taken together, these cross-dataset findings reveal that membership inference leakage is not uniform across models or data types. Instead, it emerges when models overfit, diminishes when they generalise well, and can be almost entirely suppressed through the combined use of regularisation and differential privacy. This supports the broader view that privacy risk is an emergent property of learning dynamics rather than an intrinsic

vulnerability of neural networks.

6.2 Utility–Privacy Trade-off and Practical Deployability

The experimental results clearly reveal a fundamental trade-off between model utility and privacy protection. It is important to note that **utility was only measured and evaluated within the score-based attack experiments**, where the same trained models were assessed both for predictive performance and for their susceptibility to confidence-based membership inference. The shadow and transfer attacks focused exclusively on privacy leakage, using ROC-derived metrics to isolate adversarial performance from model accuracy. This separation ensures that the privacy–utility relationship discussed here reflects the intrinsic tension between prediction quality and privacy loss, rather than differences in attack methodology.

Across all datasets, differential privacy through DP-SGD provided the strongest empirical defence, reducing attack AUCs to near-random levels. However, this came with a measurable decrease in non-member accuracy, most notably in smaller or more complex tasks. On CIFAR-10, accuracy dropped from 66.6% in the baseline to approximately 44% for the DP-SGD + regularised model at $\sigma = 1.5$, where privacy leakage became negligible. Similarly, in PathMNIST, non-member accuracy decreased from 89.4% to roughly 71% as σ increased and ε fell below 1. The same pattern appeared in RetinaMNIST, where privacy was effectively restored but overall accuracy fell to around 43%. These results confirm that privacy protection under strong noise budgets imposes a quantifiable cost on model utility.

Regularisation, in contrast, offered a more balanced outcome. Dropout and early stopping consistently reduced overfitting and improved generalisation without severe accuracy penalties. In CIFAR-10, for instance, the regularised non-private model improved non-member accuracy by nearly 5% while cutting privacy leakage by more than 20 percentage points in AUC. This indicates that conventional training stabilisation can meaningfully improve privacy resilience even in the absence of formal privacy mechanisms. When combined with DP-SGD, these methods further enhanced stability, allowing models to achieve low leakage at smaller noise multipliers, thereby retaining more utility for a given privacy budget.

From a practical perspective, the acceptable point along this privacy–utility continuum depends on the application domain. In medical imaging, where data sensitivity is paramount, the substantial reduction in leakage achieved by DP-SGD is often worth the moderate loss in accuracy. Models trained on OCTMNIST or PathMNIST maintained clinically meaningful performance while offering strong empirical privacy. Conversely, in general-purpose computer vision tasks such as CIFAR-10 or CIFAR-100, where utility is the primary design goal, applying strong DP noise may be excessive if regularisation alone can reduce leakage to near-random levels.

These findings suggest that no single mitigation strategy universally optimises both

objectives. Differential privacy guarantees formal protection at the cost of predictive accuracy, whereas regularisation provides practical mitigation with minimal degradation. In practice, the best-performing models in this study were those that combined both techniques: moderate DP noise ($\sigma \leq 1.0$) together with dropout and early stopping produced stable models with balanced privacy and utility. Such configurations align with real-world constraints in MLaaS scenarios, where slight reductions in accuracy are acceptable if they significantly diminish the risk of user data disclosure.

6.3 Evaluating ROC vs TPR@FPR Metrics

The results across all attack types reveal that the choice of evaluation metric critically shapes the interpretation of privacy leakage. In this work, privacy exposure was assessed using two complementary indicators derived from the Receiver Operating Characteristic (ROC): the *Area Under the Curve* (AUC) and the *True Positive Rate* (TPR) at fixed low *False Positive Rates* (FPR). Although both originate from the same curve, they capture fundamentally different aspects of attack success and therefore lead to distinct conclusions about the severity of privacy risk.

The AUC provides a global measure of separability between members and non-members across all possible classification thresholds. A high AUC indicates that an adversary, in principle, could distinguish between the two groups with high probability if allowed to freely choose the decision boundary. However, AUC does not account for operational constraints such as precision or acceptable error rates. In practice, an attacker cannot afford a high proportion of false positives when identifying members, especially when the underlying population is large. For this reason, privacy evaluations often emphasise performance at very low FPR levels, typically 0.1% and 1%, as these reflect realistic and high-confidence attack scenarios [26].

In contrast, TPR@FPR focuses on the adversary's practical success under strict precision requirements. It measures how many true members can be identified before exceeding a fixed, low number of false alarms. A model can exhibit a high AUC while maintaining near-zero TPR at 1% FPR, meaning that although separability exists in theory, it cannot be reliably exploited in realistic conditions. This distinction proved crucial in interpreting the experimental outcomes of this study.

Across datasets, several baseline models displayed high AUC values but very low TPR at low FPR thresholds. For example, the non-private *TanhCNN* on CIFAR-10 achieved an AUC of 72.3%, yet the corresponding TPR@1% was effectively zero. Similarly, the strong leakage observed in the CIFAR-100 baseline (AUC up to 84%) did not translate into measurable success at FPR $\leq$ 1%. These findings indicate that, although the models exposed confidence-based differences between training and unseen samples, adversaries could not leverage these differences to reliably identify individual members without incurring a large number of false positives. In operational terms, the privacy leakage was therefore limited.

The same pattern persisted across shadow and transfer attacks. Even when AUC values were moderately above 0.6 in non-private configurations, the TPR at 1% FPR remained below 0.01 in almost all cases. Once regularisation or differential privacy was applied, both metrics converged near random-guessing levels (AUC ≈ 0.5, TPR ≈ 0). This demonstrates that the defences not only reduced theoretical separability but also eliminated any meaningful exploitability.

These observations highlight the importance of reporting both global and local metrics when assessing membership inference risk. While AUC provides an upper bound on potential leakage, TPR@FPR reflects the realistic probability that a privacy breach could occur in practice. In this study, even the strongest attacks failed to achieve significant TPR at low FPR, suggesting that the evaluated models, despite measurable separability in some cases, do not present an actionable privacy threat under realistic adversarial settings. Together, these metrics confirm that differential privacy and regularisation substantially narrow the gap between member and non-member predictions, achieving both theoretical and practical resistance to membership inference.

6.4 Attack Surface Under Different Adversary Capabilities

The three membership inference attack paradigms examined in this study—score-based, shadow-model, and transfer-based—represent different levels of adversarial knowledge and access. Analysing them together provides a comprehensive understanding of how privacy leakage emerges under realistic Machine Learning as a Service (MLaaS) conditions.

The **score-based attack** assumes the strongest adversary, with full access to the model's predicted confidence scores or probability vectors. This setting is frequently used in research but is less common in deployed systems, where public APIs often expose only class labels. Because confidence outputs contain detailed information about model uncertainty, they form a direct channel through which membership signals can be detected. As expected, the score-based attack achieved the highest leakage levels across all datasets, with non-private baselines reaching up to 84% AUC on CIFAR-100 and 79% on RetinaMNIST. Once regularisation or differential privacy was applied, however, attack performance dropped to random-guessing levels (AUC ≈ 0.5, TPR@1% ≈ 0). This confirms that both defences effectively suppress the most informative leakage source.

The **shadow-model attack** represents a weaker yet more adaptive adversary. Here, the attacker trains auxiliary models on data drawn from the same distribution to approximate the behaviour of the target. This method does not require direct access to the target's confidence scores and instead relies on learning a mapping between model outputs and membership status. In this study, the shadow attacks followed the same general trend as the score-based ones. Overfitted baselines leaked strongly, while regularised or differentially private models achieved near-random performance. For example, on CIFAR-10 the non-private baseline reached an AUC of 82.3%, whereas all DP-regularised configurations

converged to 50%. This consistency across attack types shows that the privacy benefits of differential privacy generalise beyond the specific scoring mechanism used by the adversary.

The **transfer-based attack** (label-only setting) models the most restricted adversary, who observes only the predicted class labels of the target model. This scenario reflects real MLaaS deployments, where providers typically return discrete predictions rather than confidence distributions. Although earlier studies reported that membership information can sometimes be inferred from label margins or decision boundary behaviour [2, 3], the transfer attacks in this work performed near random guessing across all datasets. Even for non-private models, AUC values rarely exceeded 65%, and the true positive rate at 1% false positive rate was effectively zero. When differential privacy or regularisation was applied, leakage disappeared entirely, with ROC curves overlapping the random baseline.

Overall, the experiments reveal a clear ordering of privacy risk based on the adversary's access. Score-based attacks pose the highest risk because they exploit detailed probabilistic outputs. Shadow-model attacks can reproduce similar leakage if auxiliary data are available, but their success still depends on the generalisation gap of the target. Transfer attacks, which rely only on predicted labels, show negligible success and pose little practical threat under normal service constraints. Despite these differences, all three attack types converged to random-guessing behaviour once differential privacy and regularisation were combined. This demonstrates that the evaluated defences provide consistent and robust protection even against the strongest feasible adversary in MLaaS settings.

6.5 Influence of Model Capacity, Pretraining, and Data Regime

Model architecture and data characteristics play a decisive role in determining both the magnitude of privacy leakage and the effectiveness of mitigation strategies. The results demonstrate that the same defence mechanisms behave differently depending on the network's capacity, the use of pretraining, and the underlying dataset complexity.

High-capacity models trained from scratch, such as the non-pretrained *ResNet-18* and *WideResNet-28-10* on CIFAR-100, displayed the strongest overfitting and consequently the highest privacy leakage. These models achieved nearly perfect accuracy on their training sets but failed to generalise to unseen data, resulting in wide confidence gaps that enabled successful membership inference. This behaviour is consistent with the understanding that deep networks with large parameter counts tend to memorise training samples when the dataset size is limited or regularisation is insufficient [3].

By contrast, the ImageNet-pretrained *ResNet-18* on CIFAR-100 exhibited almost no measurable leakage, even without differential privacy. Despite operating on the same data distribution, its AUC remained close to 0.5 across all regimes. This outcome indicates that transfer learning inherently stabilises training by providing strong, generalisable feature representations. Because the pretrained backbone already captures domain-agnostic patterns, the fine-tuning process requires less adaptation to individual samples, thereby

reducing memorisation and mitigating membership risk. These findings suggest that pre-training can act as an implicit regulariser that naturally enhances privacy without explicit noise injection.

Similar trends were observed across the medical datasets. In OCTMNIST and PathMNIST, the architectures maintained high accuracy and low leakage, which can be attributed to the moderate model capacity and the limited visual diversity of the datasets. In contrast, RetinaMNIST exhibited the highest leakage among the medical tasks, reflecting its smaller size and higher label noise. These characteristics make the model more prone to overfitting and amplify the impact of memorised samples on the decision boundary. Although differential privacy successfully reduced this leakage, it also degraded accuracy sharply, showing that data-scarce domains face a narrower privacy–utility margin.

Taken together, the results highlight that privacy leakage is a product of the interaction between model capacity, training dynamics, and data complexity rather than an inherent weakness of neural networks. Large models trained from scratch on limited data are more likely to memorise specific examples and therefore reveal membership information. Pre-trained architectures, on the other hand, benefit from a stable feature space that generalises better and leaks less, even in the absence of explicit privacy constraints. These observations reinforce the idea that architectural and data-centric choices can complement formal privacy mechanisms, providing a practical foundation for privacy-aware model design.

6.6 Limitations and Threat Model Gaps

While the experimental findings provide a consistent picture of how differential privacy and regularisation mitigate membership inference, several methodological limitations and threat model assumptions must be acknowledged. These constraints define the scope of the results and highlight potential directions for future work.

First, all attacks were conducted under a *black-box* setting, where the adversary interacts with the model only through its predictions. This reflects realistic MLaaS conditions but excludes stronger *white-box* adversaries that could exploit gradients, activations, or training checkpoints to recover sensitive information. Extending the evaluation to white-box or hybrid-access settings would provide a more complete understanding of model vulnerability.

Second, the analysis focused on three primary attack paradigms: score-based, shadow-model, and transfer-based. Although these cover a wide range of adversarial capabilities, they do not represent the full spectrum of possible membership inference strategies. Other attack formulations that adapt dynamically to model confidence distributions or exploit auxiliary information may achieve stronger results and could be explored in future work.

Third, all differentially private models were trained using the Opacus implementation of DP-SGD with Gaussian noise and a single-accountant framework for privacy budget computation. While this approach offers a well-established and reproducible baseline, it does not capture the full variety of differential privacy mechanisms or accounting tech-

niques. Alternative methods, such as Rényi differential privacy or per-layer clipping adjustments, may yield different privacy–utility trade-offs and deserve further investigation.

Fourth, each configuration was evaluated using a single run of the final checkpoint. Although care was taken to control random seeds and ensure reproducible training conditions, the inherent stochasticity of both SGD and DP noise introduces run-to-run variability that can influence AUC and TPR results. Repeating experiments and reporting aggregated statistics would increase the robustness and generalisability of the conclusions.

Finally, the experiments were limited to image classification datasets of moderate size with balanced label distributions. Privacy behaviour may differ in other modalities, such as text, tabular, or multimodal data, where overfitting dynamics and representational biases differ significantly. The relationship between dataset diversity, model scale, and membership leakage therefore warrants additional empirical study.

Recognising these limitations helps situate the findings within a realistic experimental scope. The observed trends of reduced leakage through regularisation and differential privacy remain consistent across all datasets and attack types. Expanding the threat model and experimental range would enable a more complete understanding of how these defences perform in larger and more complex settings.. The observed trends—reduced leakage through regularisation and differential privacy—remain consistent across all tested datasets and attack types. Expanding the threat model and experimental scope would enable a more comprehensive understanding of how these defences scale to larger architectures and more complex adversarial conditions.

6.7 Implications for MLaaS and Future Work

The results of this study have direct implications for the design and deployment of Machine Learning as a Service (MLaaS) systems. They show that the extent of membership inference risk depends strongly on how models are trained, what outputs are exposed to users, and which privacy mechanisms are applied. In practice, privacy protection should be approached as a design decision that balances predictive performance, interpretability, and user trust.

From a deployment perspective, the experiments confirm that exposing detailed confidence scores or probability distributions greatly increases privacy risk. APIs that return only discrete class labels, as seen in the transfer-based attack evaluations, provide a significantly safer interface. This finding suggests that privacy-by-design principles can be implemented not only through algorithmic defences but also through careful control of model outputs. Limiting access to softmax probabilities or intermediate activations can effectively reduce the attack surface without requiring changes to the underlying training process.

At the algorithmic level, the combined use of differential privacy and regularisation proved to be the most reliable mitigation strategy. Moderate DP noise, applied together with dropout and early stopping, achieved strong empirical privacy while maintaining

acceptable accuracy. Such configurations are particularly suitable for medical and high-sensitivity applications, where the cost of reduced utility is outweighed by the benefit of stronger data protection. For large-scale consumer models where utility is paramount, standard regularisation may already be sufficient to keep membership risk near random levels. This suggests that privacy-preserving model training should be context-dependent rather than uniform across all applications.

The findings also highlight the importance of transparency in communicating privacy guarantees to users of MLaaS platforms. Reporting privacy budgets, regularisation settings, and validation performance can help service providers demonstrate compliance with principles such as data minimisation and privacy by design, as described in the General Data Protection Regulation (GDPR). Such transparency strengthens user confidence and promotes responsible model deployment.

Future research should extend these experiments in three main directions. First, larger and more diverse datasets should be examined to evaluate how privacy leakage scales with model capacity and domain complexity. Second, investigations could include adaptive adversaries and gradient-based access models to bridge the gap between black-box and white-box settings. Third, integrating privacy auditing tools and memorisation metrics would allow for continuous monitoring of leakage risk in deployed systems.

In summary, the empirical evidence presented in this thesis suggests that privacy-aware design is both achievable and practical within MLaaS frameworks. Restricting output access, combining differential privacy with regularisation, and ensuring transparent reporting can jointly provide strong protection against membership inference while preserving the essential utility of modern machine learning models.

Chapter 7

Conclusion

This chapter concludes the thesis by integrating the experimental findings and theoretical insights into a unified perspective on privacy risks in machine learning. It revisits the core research aim, summarises the main contributions, and provides direct answers to the guiding research questions. The discussion here moves from detailed analysis toward synthesis, outlining what the results collectively reveal about privacy leakage, model behaviour, and practical defence strategies.

The overarching goal of this work was to examine how machine learning models trained on both natural and medical image datasets are affected by membership inference attacks and how different defence mechanisms influence this vulnerability. The study explored the interplay between model generalisation, regularisation, and differential privacy, assessing how these factors jointly determine the trade-off between utility and privacy.

To achieve this goal, models were trained under four regimes: standard non-private, regularised (with dropout and early stopping), differentially private (DP-SGD), and a combined DP-SGD + regularised configuration. Privacy was evaluated under three black-box adversarial settings representing increasing restrictions of access and realism: a score-based attack using output confidences, a shadow-model attack leveraging auxiliary data, and a transfer-based attack operating on label-only outputs. Together, these experiments provided a comprehensive assessment of privacy leakage across both natural and medical image domains.

The following sections summarise the key findings and present concise answers to the three research questions formulated in Chapter 1:

1. **RQ1:** How do different privacy attacks perform against machine learning models trained on medical and natural image datasets with varying levels of sensitivity and structure?

2. **RQ2:** What is the extent of privacy leakage due to membership inference in such models, and which metric is more appropriate for quantifying this leakage?

3. **RQ3:** Which mitigation strategies can effectively reduce privacy leakage while maintaining acceptable model utility?

By addressing these questions, this chapter summarises how adversarial strength, model design, and dataset characteristics jointly shape the privacy landscape of modern image-based machine learning systems.

7.1 Answers to the Research Questions

RQ1: Performance of Different Privacy Attacks Across Datasets

The three evaluated attack paradigms revealed a consistent hierarchy of effectiveness across both natural and medical image datasets. Score-based attacks, which rely on model confidence scores, were the most capable of identifying training samples because they exploit detailed information in the output probabilities. Shadow-model attacks, which train auxiliary models on data drawn from the same distribution, reproduced similar leakage patterns but achieved slightly lower accuracy. Transfer-based attacks, which operate only on predicted class labels, were the least effective and typically indistinguishable from random guessing.

Across all datasets, leakage appeared strongest in overfitted non-private models and weakest in regularised or differentially private configurations. Natural image datasets such as CIFAR-10 and CIFAR-100 exhibited greater susceptibility to membership inference due to their complexity and the higher capacity of the networks trained on them. Medical datasets showed weaker leakage overall, particularly in OCTMNIST and PathMNIST, where models generalised well and produced fewer membership cues. These observations demonstrate that adversarial strength, model exposure, and dataset structure jointly determine the extent of privacy risk.

RQ2: Extent and Measurement of Privacy Leakage

Privacy leakage was directly related to the degree of overfitting observed during training. Models with large gaps between training and test accuracy enabled stronger attacks, while well-regularised and differentially private models produced nearly indistinguishable outputs for members and non-members. Leakage was quantified using two complementary metrics: the Area Under the ROC Curve (AUC) and the True Positive Rate (TPR) at fixed low False Positive Rates (FPR). AUC captured global separability between the two classes, whereas TPR@FPR reflected the adversary's success under realistic precision constraints. Several models achieved moderately high AUC values yet retained near-zero TPR at 1% FPR, indicating that apparent separability rarely translated into effective exploitation. Together, these metrics provided a complete picture of both theoretical and practical privacy risk.

RQ3: Effective Mitigation Strategies for Privacy Leakage

Differential privacy and regularisation each reduced membership inference risk, although with different trade-offs. Differential privacy, implemented through DP-SGD, enforced stability in the learning process and achieved the strongest overall protection. When the noise multiplier σ exceeded 1.0, AUC values approached 0.5 and TPR at 1% FPR fell to zero across all datasets, signalling near-complete resistance to attack. This level of privacy, however, required accepting lower model accuracy, particularly for smaller or noisier datasets such as RetinaMNIST.

Regularisation through dropout and early stopping achieved weaker but more efficient protection by reducing overfitting without severely impacting performance. Combining these techniques with DP-SGD allowed models to reach comparable privacy levels at lower noise multipliers, preserving higher accuracy for a given privacy budget. This joint strategy offers a practical balance between privacy and utility for real-world MLaaS deployments, providing strong empirical protection while maintaining functional predictive capability.

7.2 Contributions and Significance

This work makes several contributions to the empirical study of privacy in machine learning. It advances understanding of how membership inference attacks interact with model design, data characteristics, and training regimes, and it provides evidence for effective defence strategies that balance utility and protection.

Comprehensive cross-domain evaluation. The study presents a unified analysis of membership inference across both natural and medical image datasets. By applying the same experimental framework to CIFAR-10, CIFAR-100, OCTMNIST, RetinaMNIST, and PathMNIST, it captures how dataset structure and sensitivity influence leakage behaviour. This comparison extends prior work that typically focused on either generic benchmarks or a single application domain.

Systematic comparison of attack paradigms. Three black-box attack classes were implemented and evaluated under identical conditions: score-based, shadow-model, and transfer-based. Their relative performance establishes a clear hierarchy of adversarial strength and demonstrates that privacy risk diminishes rapidly as model access becomes more limited. This provides practical insight into which threat models are realistic for MLaaS deployments.

Empirical analysis of the privacy–utility relationship. By measuring non-member accuracy and privacy metrics side by side, the experiments quantify how defences alter the balance between predictive performance and resistance to inference. The results show that differential privacy and regularisation converge toward the same goal of reducing overfitting, although through different mechanisms. Their combination proved to be the most consistent way to maintain generalisation while eliminating measurable leakage.

Clarification of evaluation metrics. The distinction between the Area Under the ROC Curve and the True Positive Rate at fixed low False Positive Rates is analysed both theoretically and empirically. This dual-metric view highlights that high AUC values do not necessarily indicate practical vulnerability when TPR at low FPR remains near zero. It establishes a more accurate and operational definition of privacy risk for future studies.

Guidance for privacy-aware model design. The findings demonstrate that moderate DP noise combined with dropout and early stopping provides strong empirical protection with tolerable accuracy loss. This approach offers a realistic blueprint for privacy-preserving training in MLaaS environments, where service providers must balance regulatory compliance, computational efficiency, and model performance.

Together, these contributions advance the empirical understanding of membership inference and provide actionable insights for designing, evaluating, and deploying privacy-resilient machine learning models.

7.3 Closing Reflection

The investigation conducted in this thesis demonstrates that privacy and generalisation are deeply connected aspects of modern machine learning. Through a unified evaluation of multiple attack paradigms, datasets, and defence strategies, it becomes clear that models which generalise well also protect their training data more effectively. Regularisation and differential privacy each contribute to this stability by limiting memorisation and promoting robustness to small perturbations. The findings therefore support the view that privacy should not be treated as an external constraint added after training, but as a property that emerges from sound model design and optimisation. Continued exploration of this relationship between privacy, learning dynamics, and generalisation will be essential for building machine learning systems that are both effective and trustworthy in practice.

In this context, the guiding philosophy of *Privacy by Design* provides a valuable perspective. It seeks to accommodate all legitimate interests and objectives in a positive-sum, "win–win" manner, rather than through a dated, zero-sum approach where unnecessary trade-offs are made. Privacy by Design avoids the pretence of false dichotomies such as privacy versus security, demonstrating that it is both possible and preferable to achieve strong privacy protection and high system performance simultaneously[38].

Bibliography

[1] IBM. *The Not So Short A Introduction to LaTeX2e*. https://www.ibm.com/topics/machine-learning.

[2] Reza Shokri et al. "Membership Inference Attacks Against Machine Learning Models". eng. In: *2017 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2017, pp. 3–18. ISBN: 9781509055326.

[3] Hongsheng Hu et al. "Membership Inference Attacks on Machine Learning: A Survey". In: 54.11s (2022). ISSN: 0360-0300. DOI: 10.1145/3523273. URL: https://doi.org/10.1145/3523273.

[4] Ahmed Salem et al. *ML-Leaks: Model and Data Independent Membership Inference Attacks and Defenses on Machine Learning Models*. 2018. arXiv: 1806.01246 [cs.CR]. URL: https://arxiv.org/abs/1806.01246.

[5] Zheng Li and Yang Zhang. "Membership Leakage in Label-Only Exposures". In: *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security (CCS)*. 2021, pp. 880–895. DOI: 10.1145/3460120.3484575. URL: https://dl.acm.org/doi/10.1145/3460120.3484575.

[6] Yuefeng Peng et al. "OSLO: One-Shot Label-Only Membership Inference Attacks". In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2024. URL: https://proceedings.neurips.cc/paper_files/paper/2024/file/71f88122d414cfeb455ac0ed932fbe1f-Paper-Conference.pdf.

[7] Nicholas Carlini et al. "Membership Inference Attacks from First Principles". In: *2022 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2022, pp. 1897–1914. URL: https://arxiv.org/abs/2112.03570.

[8] Sasi Kumar Murakonda and Reza Shokri. "ML Privacy Meter: Aiding Regulatory Compliance by Quantifying the Privacy Risks of Machine Learning". In: *arXiv preprint arXiv:2007.09339* (2020). Presented at HotPETs 2020. URL: https://arxiv.org/abs/2007.09339.

[9] T. Ross. "The Synthesis of Intelligence–its Implications". In: *Psychological Review* 45.2 (1938), pp. 185–189. DOI: 10.1037/h0059815.

[10] A. L. Samuel. “Some Studies in Machine Learning Using the Game of Checkers”. In: *IBM Journal of Research and Development* 3.3 (1959), pp. 210–229. DOI: 10.1147/rd.33.0210.

[11] Gareth James et al. *An Introduction to Statistical Learning: with Applications in R*. Springer, 2013. URL: https://faculty.marshall.usc.edu/gareth-james/ISL/.

[12] Vineeth Balasubramanian, shen shyang ho shen shyang, and Vladimir Vovk. “Conformal Prediction for Reliable Machine Learning: Theory, Adaptations and Applications”. In: *Conformal Prediction for Reliable Machine Learning: Theory, Adaptations and Applications* (Apr. 2014).

[13] S. De Backer, A. Naud, and P. Scheunders. “Non-linear dimensionality reduction techniques for unsupervised feature extraction”. In: *Pattern Recogn. Lett.* 19.8 (June 1998), pp. 711–720. ISSN: 0167-8655. DOI: 10.1016/S0167-8655(98)00049-X. URL: https://doi.org/10.1016/S0167-8655(98)00049-X.

[14] Heeyoul "Henry Choi and Seungjin Choi. “Robust kernel Isomap”. In: *Pattern Recognition* 40 (Mar. 2007), pp. 853–862. DOI: 10.1016/j.patcog.2006.04.025.

[15] Aurelien Geron. *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*. 2nd. O’Reilly Media, Inc., 2019. ISBN: 1492032646.

[16] Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. 3rd ed. Prentice Hall, 2010.

[17] Diederik Kingma and Jimmy Ba. “Adam: A Method for Stochastic Optimization”. In: *International Conference on Learning Representations* (Dec. 2014).

[18] F. Rosenblatt. “The perceptron: A probabilistic model for information storage and organization in the brain [J]”. In: *Psychol. Review* 65 (Nov. 1958), pp. 386–408. DOI: 10.1037/h0042519.

[19] Marvin Minsky and Seymour Papert. *Perceptrons: An Introduction to Computational Geometry*. Sept. 2017. ISBN: 9780262343930. DOI: 10.7551/mitpress/11301.001.0001.

[20] Prajit Ramachandran, Barret Zoph, and Quoc V. Le. *Searching for Activation Functions*. 2017. arXiv: 1710.05941 [cs.NE]. URL: https://arxiv.org/abs/1710.05941.

[21] Cynthia Dwork et al. “Calibrating Noise to Sensitivity in Private Data Analysis”. In: *Proceedings of the 3rd Theory of Cryptography Conference (TCC)*. Springer, 2006, pp. 265–284. DOI: 10.1007/11681878_14.

[22] Samuel Yeom et al. *Privacy Risk in Machine Learning: Analyzing the Connection to Overfitting*. 2018. arXiv: 1709.01604 [cs.CR]. URL: https://arxiv.org/abs/1709.01604.

[23] Matt Fredrikson, Somesh Jha, and Thomas Ristenpart. "Model Inversion Attacks that Exploit Confidence Information and Basic Countermeasures". In: *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*. CCS '15. Denver, Colorado, USA: Association for Computing Machinery, 2015, pp. 1322–1333. ISBN: 9781450338325. DOI: 10.1145/2810103.2813677. URL: https://doi.org/10.1145/2810103.2813677.

[24] Giuseppe Ateniese et al. "Hacking smart machines with smarter ones: How to extract meaningful data from machine learning classifiers". In: *Int. J. Secur. Netw.* 10.3 (Sept. 2015), pp. 137–150. ISSN: 1747-8405. DOI: 10.1504/IJSN.2015.071829. URL: https://doi.org/10.1504/IJSN.2015.071829.

[25] Luca Melis et al. *Exploiting Unintended Feature Leakage in Collaborative Learning*. 2018. arXiv: 1805.04049 [cs.CR]. URL: https://arxiv.org/abs/1805.04049.

[26] Nicholas Carlini et al. *Extracting Training Data from Large Language Models*. 2021. arXiv: 2012.07805 [cs.CR]. URL: https://arxiv.org/abs/2012.07805.

[27] Milad Nasr, Reza Shokri, and Amir Houmansadr. "Comprehensive Privacy Analysis of Deep Learning: Passive and Active White-box Inference Attacks against Centralized and Federated Learning". In: *2019 IEEE Symposium on Security and Privacy (SP)*. IEEE, May 2019, pp. 739–753. DOI: 10.1109/sp.2019.00065. URL: http://dx.doi.org/10.1109/SP.2019.00065.

[28] Martin Abadi et al. "Deep Learning with Differential Privacy". In: *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. CCS'16. ACM, Oct. 2016, pp. 308–318. DOI: 10.1145/2976749.2978318. URL: http://dx.doi.org/10.1145/2976749.2978318.

[29] Keith Bonawitz et al. "Practical Secure Aggregation for Privacy-Preserving Machine Learning". In: *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. CCS '17. Dallas, Texas, USA: Association for Computing Machinery, 2017, pp. 1175–1191. ISBN: 9781450349468. DOI: 10.1145/3133956.3133982. URL: https://doi.org/10.1145/3133956.3133982.

[30] Stacey Truex et al. "Demystifying Membership Inference Attacks in Machine Learning as a Service". In: *IEEE Transactions on Services Computing* 14.6 (2021), pp. 2073–2089. DOI: 10.1109/TSC.2019.2897554.

[31] Klas Leino and Matt Fredrikson. "Stolen Memories: Leveraging Model Memorization for Calibrated White-Box Membership Inference". In: *29th USENIX Security Symposium (USENIX Security 20)*. USENIX Association, Aug. 2020, pp. 1605–1622. ISBN: 978-1-939133-17-5. URL: https://www.usenix.org/conference/usenixsecurity20/presentation/leino.

[32] Hongbin Liu et al. "EncoderMI: Membership Inference against Pre-trained Encoders in Contrastive Learning". In: *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*. CCS '21. Virtual Event, Republic of Korea: Association for Computing Machinery, 2021, pp. 2081–2095. ISBN: 9781450384544. DOI: 10.1145/3460120.3484749. URL: https://doi.org/10.1145/3460120.3484749.

[33] Christopher A. Choquette-Choo et al. "Label-Only Membership Inference Attacks". In: *Proceedings of the 38th International Conference on Machine Learning*. Ed. by Marina Meila and Tong Zhang. Vol. 139. Proceedings of Machine Learning Research. PMLR, 18–24 Jul 2021, pp. 1964–1974. URL: https://proceedings.mlr.press/v139/choquette-choo21a.html.

[34] Zheng Li and Yang Zhang. "Membership Leakage in Label-Only Exposures". In: *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*. CCS '21. Virtual Event, Republic of Korea: Association for Computing Machinery, 2021, pp. 880–895. ISBN: 9781450384544. DOI: 10.1145/3460120.3484575. URL: https://doi.org/10.1145/3460120.3484575.

[35] Xiao Li et al. *On the Privacy Effect of Data Enhancement via the Lens of Memorization*. 2024. arXiv: 2208.08270 [cs.LG]. URL: https://arxiv.org/abs/2208.08270.

[36] Jia Deng et al. "ImageNet: A large-scale hierarchical image database". In: *2009 IEEE Conference on Computer Vision and Pattern Recognition*. 2009, pp. 248–255. DOI: 10.1109/CVPR.2009.5206848.

[37] Jiancheng Yang, Rui Shi, and Bingbing Ni. "MedMNIST Classification Decathlon: A Lightweight AutoML Benchmark for Medical Image Analysis". In: *2021 IEEE 18th International Symposium on Biomedical Imaging (ISBI)*. 2021, pp. 191–195. DOI: 10.1109/ISBI48211.2021.9434062.

[38] Ann Cavoukian. "Privacy by design: the definitive workshop. A foreword by Ann Cavoukian, Ph.D". In: *Identity in the Information Society* 3 (Aug. 2010), pp. 247–251. DOI: 10.1007/s12394-010-0062-y.

Appendix A

Supplementary figures and notebooks

This appendix provides the complete set of visual and diagnostic results for every model configuration discussed in Chapter 5. Each dataset section contains one subsection per training regime, including the baseline (no dropout, no early stopping), the regularised model, the DP-SGD model, and the DP-SGD model with regularisation. For each configuration, the following figures are included:

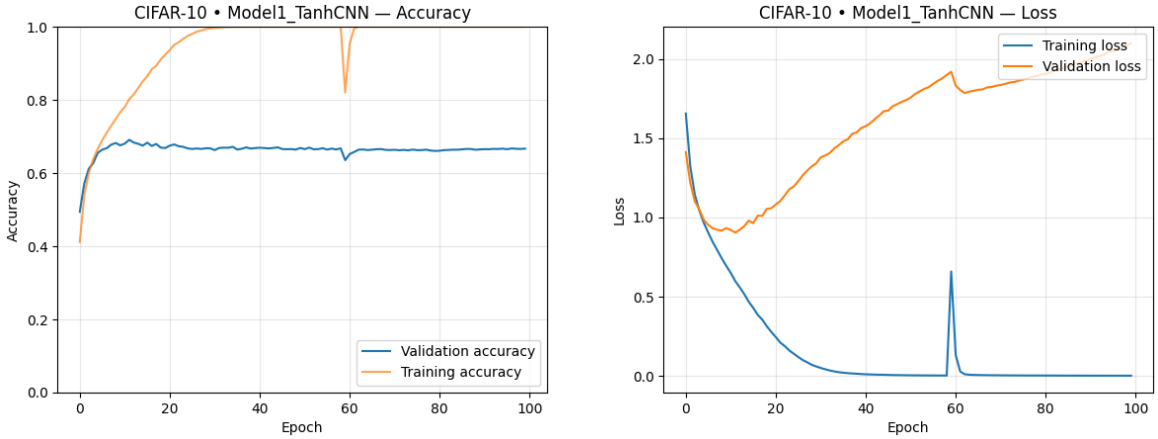
- Training and validation accuracy/loss curves across epochs.
- ROC curve for the confidence-based membership inference attack.
- Confidence distributions comparing members and non-members.
- Accuracy comparison between members and non-members.

Where relevant, a link to the corresponding interactive Colab notebook is also provided for full reproducibility. All figures use data from the final run of each configuration, as reported in Chapter 5.

A.1 CIFAR-10

A.1.1 Baseline (no dropout, no early stopping)

LINK: `https://colab.research.google.com/drive/1iu9UQR1y-WN5WRPRuglgEA-dmsCJjKDk?usp=sharing`



(a) Accuracy. **(b)** Loss.

Figure A.1: Training and validation performance for the CIFAR-10 baseline model.

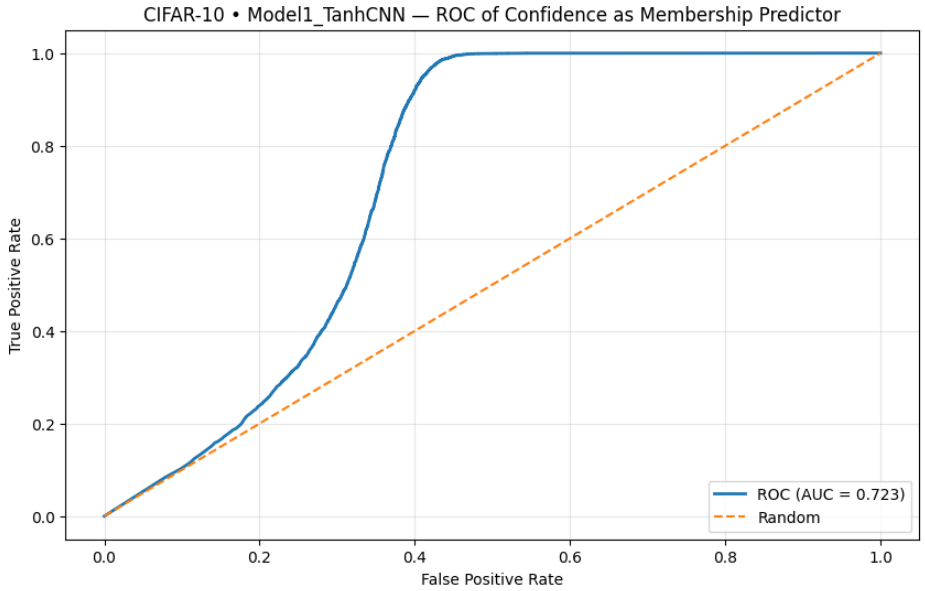


Figure A.2: ROC curve for confidence-based MIA (CIFAR-10 baseline).

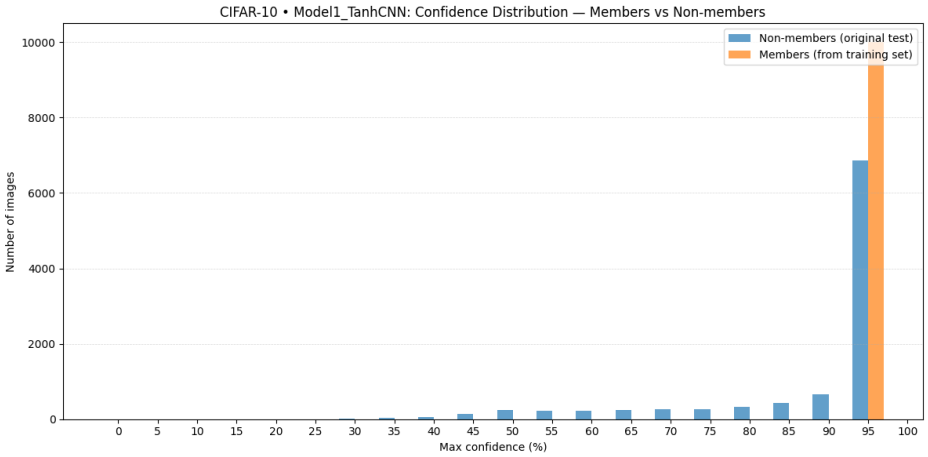


Figure A.3: Confidence distributions for members vs non-members (CIFAR-10 baseline).

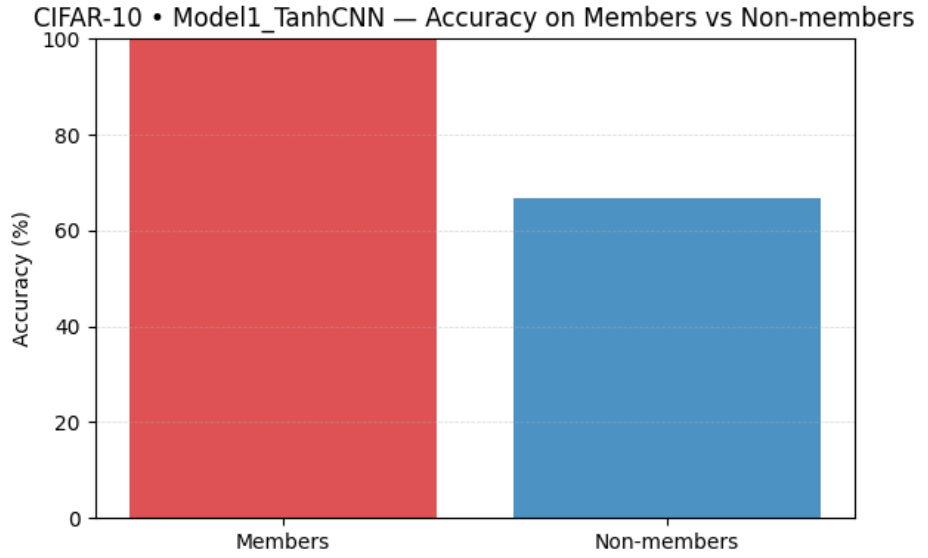
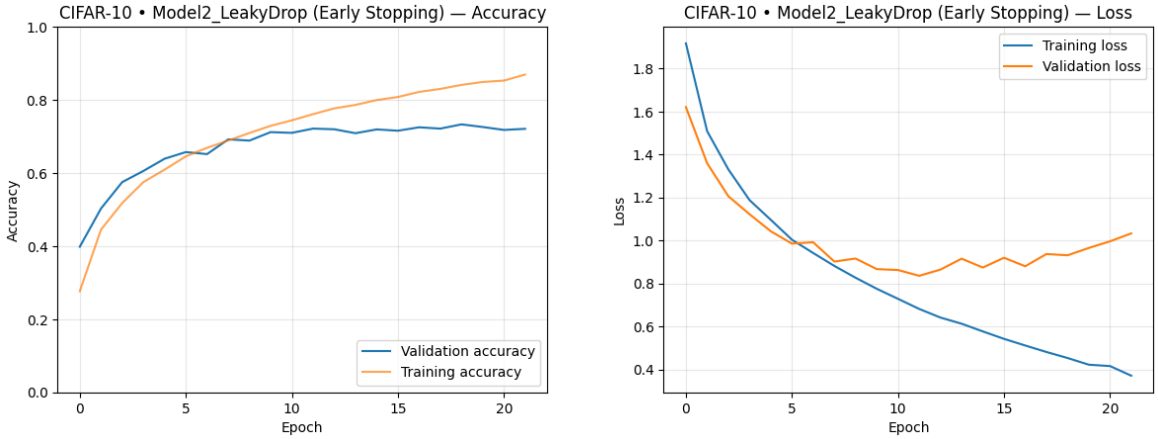


Figure A.4: Accuracy comparison for members and non-members (CIFAR-10 baseline).

A.1.2 Regularised (dropout + early stopping)

Link: `https://colab.research.google.com/drive/1Qc9XARafGAbIBCFxnawFVPWb00YR60JJ?usp=sharing`



(a) Accuracy.

(b) Loss.

Figure A.5: Training and validation performance for the CIFAR-10 Regularised model.

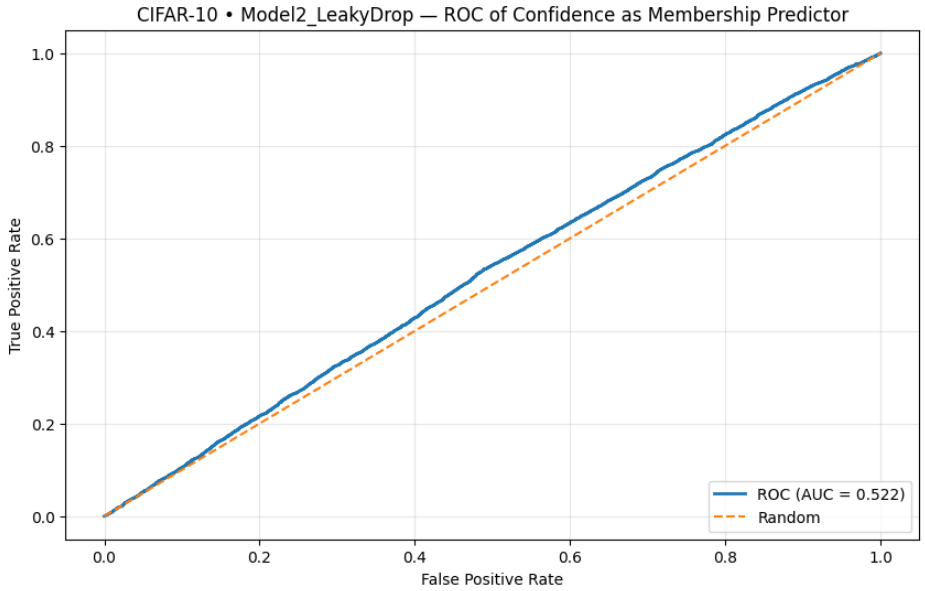


Figure A.6: ROC curve for confidence-based MIA (CIFAR-10 Regularised).

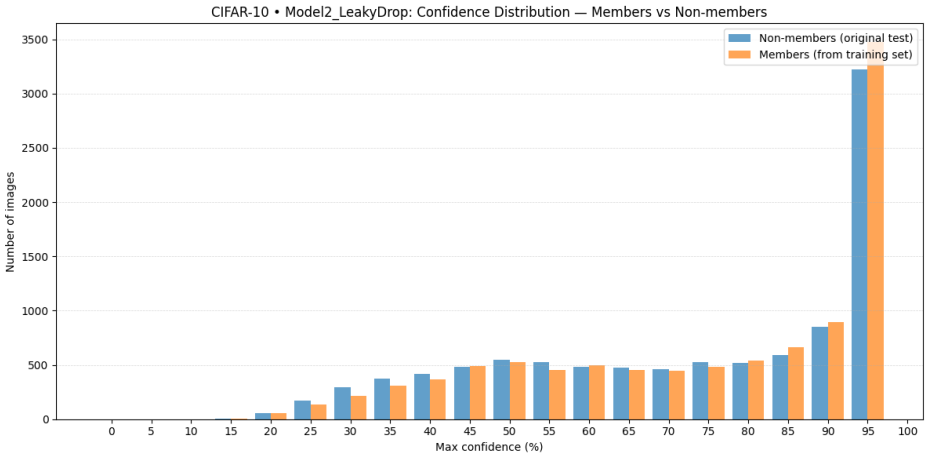


Figure A.7: Confidence distributions for members vs non-members (CIFAR-10 Regularised.

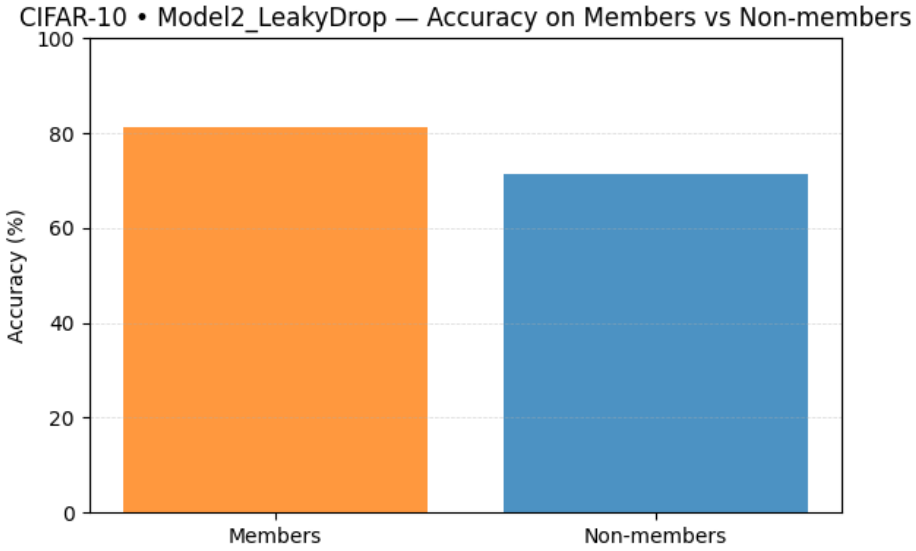
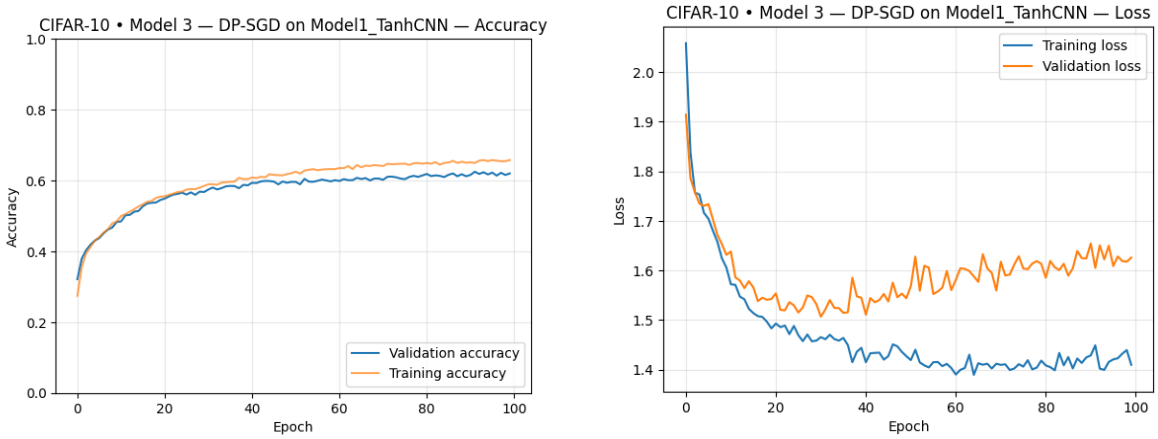


Figure A.8: Accuracy comparison for members and non-members (CIFAR-10 Regularised).

A.1.3 DP-SGD (TanhCNN)

Link: `https://colab.research.google.com/drive/1mNV_J7ep39MhNE5-X14RIpGKQ5HoVmvR?usp=sharing`

Epsilon ε = 0.5



(a) Accuracy. **(b)** Loss.

Figure A.9: Training and validation performance for the CIFAR-10 DP-No-Regularised with ε = 0.5.

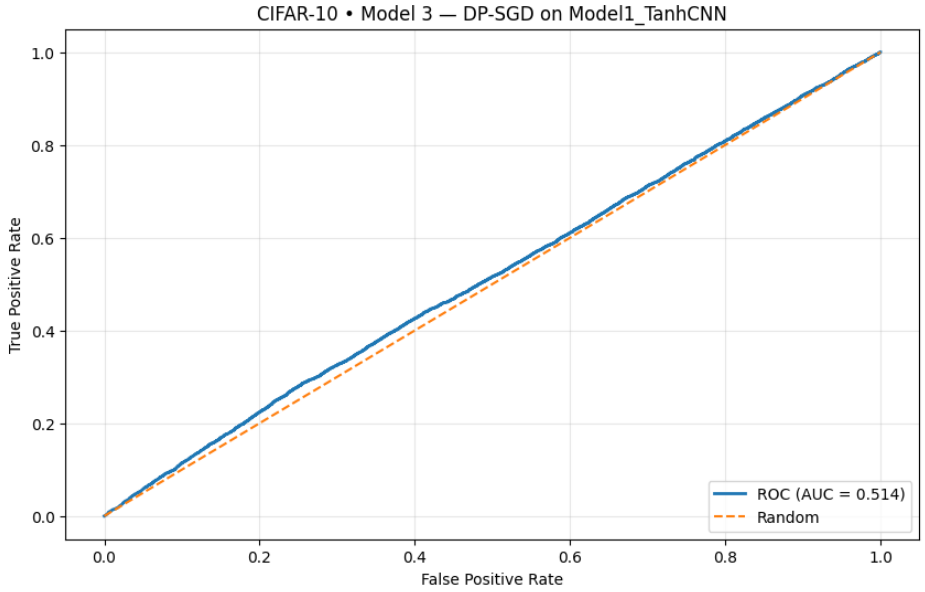


Figure A.10: ROC curve for confidence-based MIA (CIFAR-10 DP-No-Regularised with ε = 0.5).

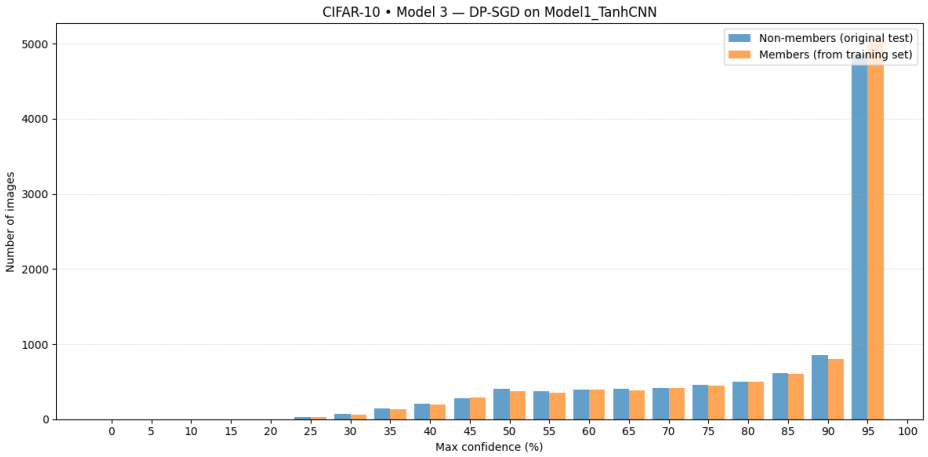


Figure A.11: Confidence distributions for members vs non-members (CIFAR-10 DP-No-Regularised with ε = 0.5.

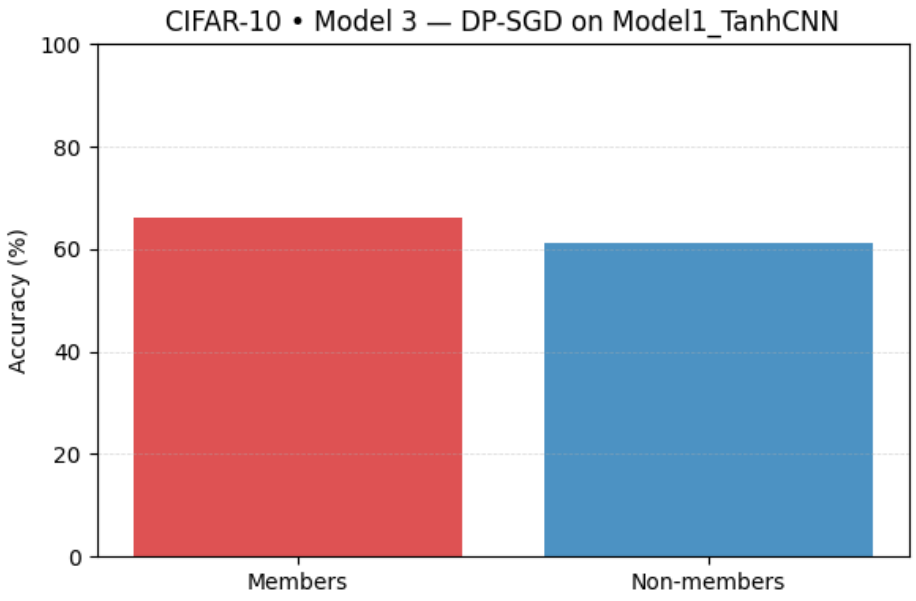
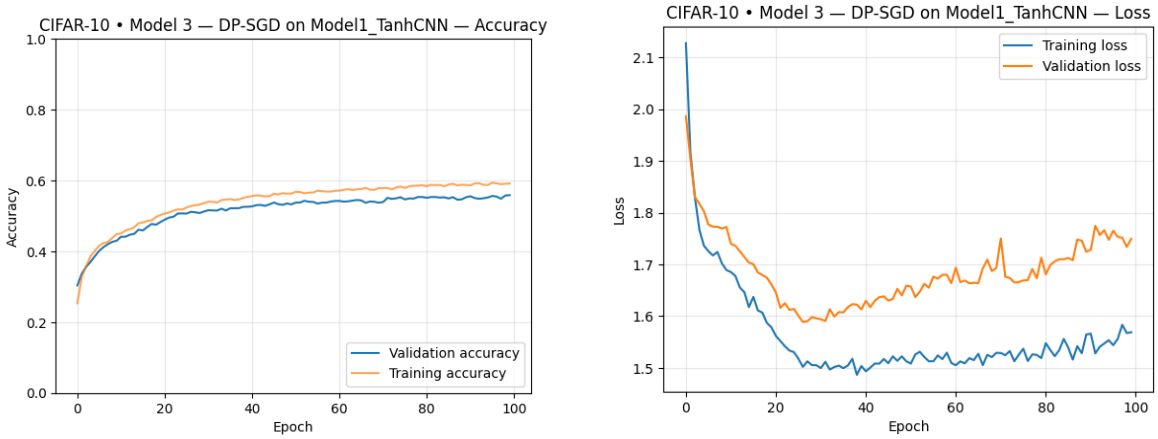


Figure A.12: Accuracy comparison for members and non-members CIFAR-10 DP-No-Regularised with ε = 0.5).

Epsilon ε = 1.0



(a) Accuracy. **(b)** Loss.

Figure A.13: Training and validation performance for the CIFAR-10 DP-No-Regularised with ε = 1.0.

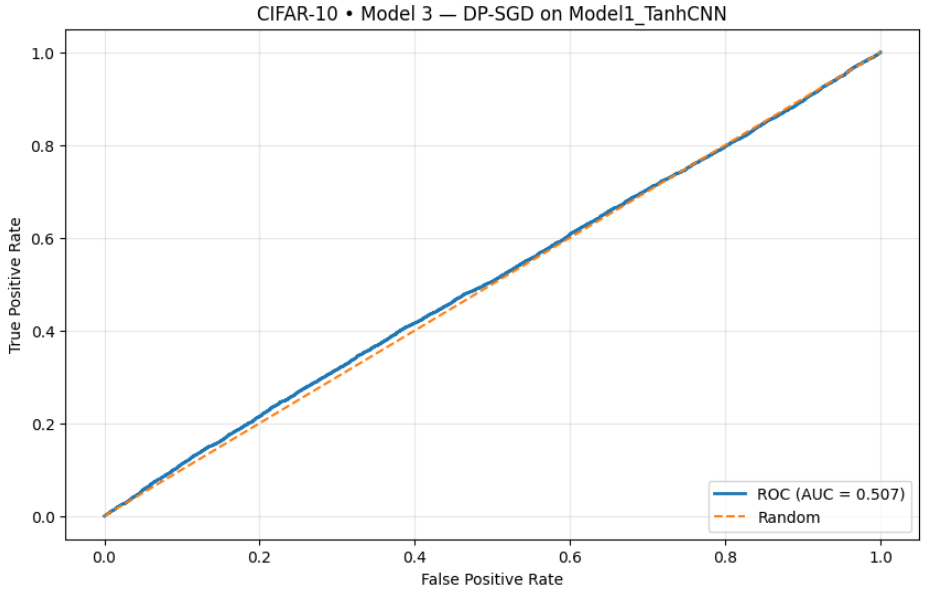


Figure A.14: ROC curve for confidence-based MIA (CIFAR-10 DP-No-Regularised with ε = 1.0).

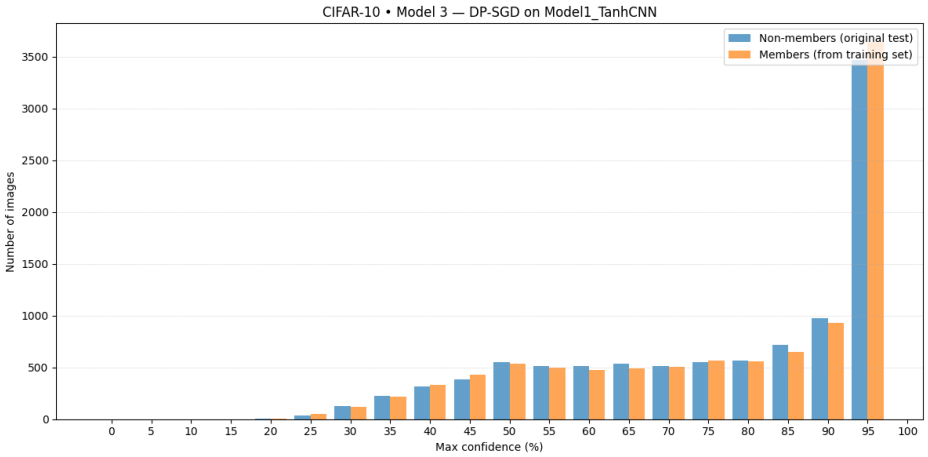


Figure A.15: Confidence distributions for members vs non-members (CIFAR-10 DP-No-Regularised with ε = 0.5.

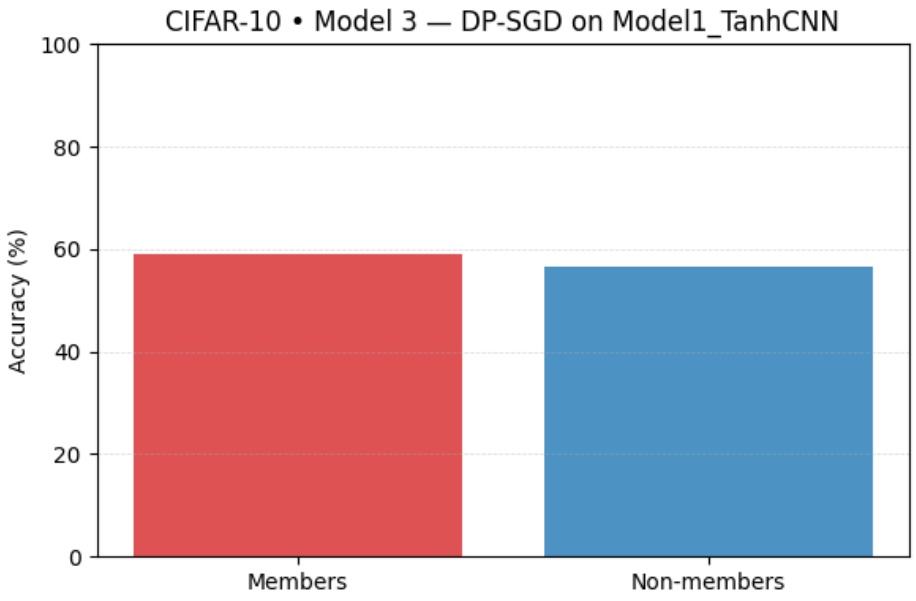
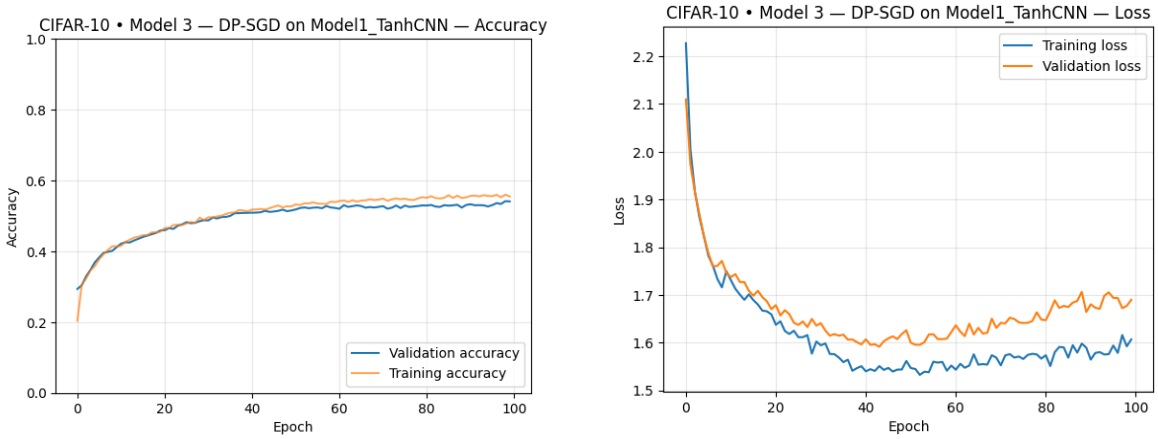


Figure A.16: Accuracy comparison for members and non-members CIFAR-10 DP-No-Regularised with ε = 1.0).

Epsilon ε = 1.5



(a) Accuracy. **(b)** Loss.

Figure A.17: Training and validation performance for the CIFAR-10 DP-No-Regularised with ε = 1.5.

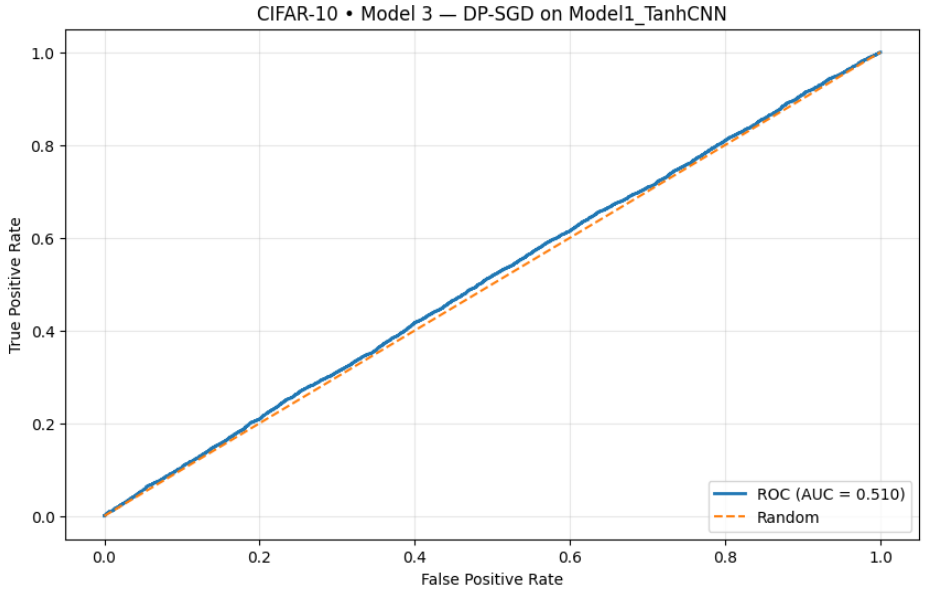


Figure A.18: ROC curve for confidence-based MIA (CIFAR-10 DP-No-Regularised with ε = 1.5).

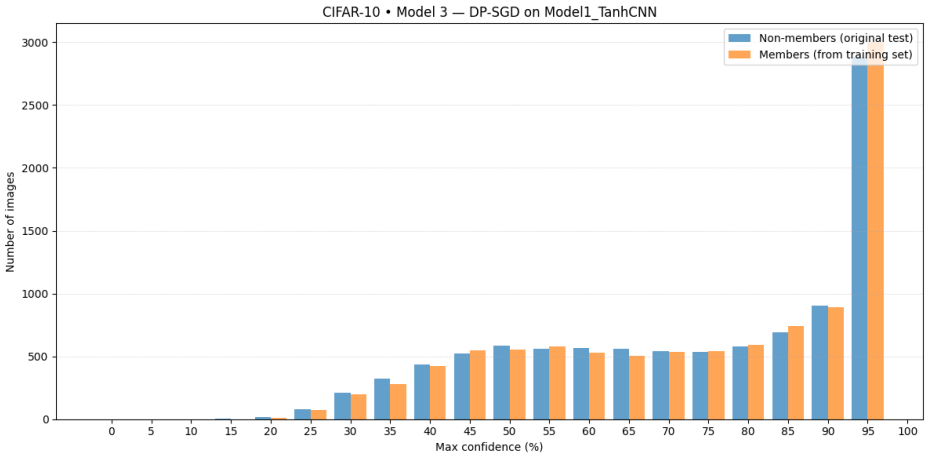


Figure A.19: Confidence distributions for members vs non-members (CIFAR-10 DP-No-Regularised with ε = 1.5.

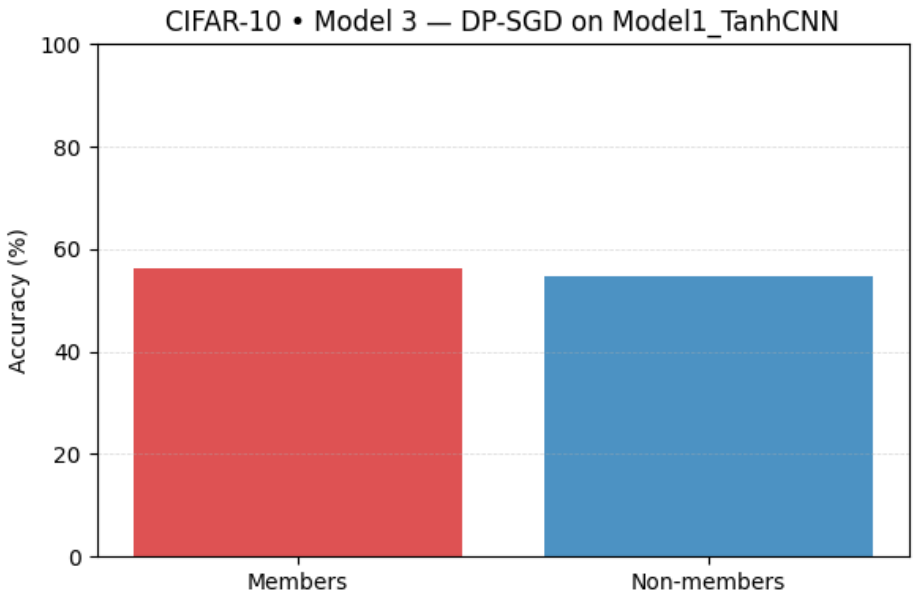
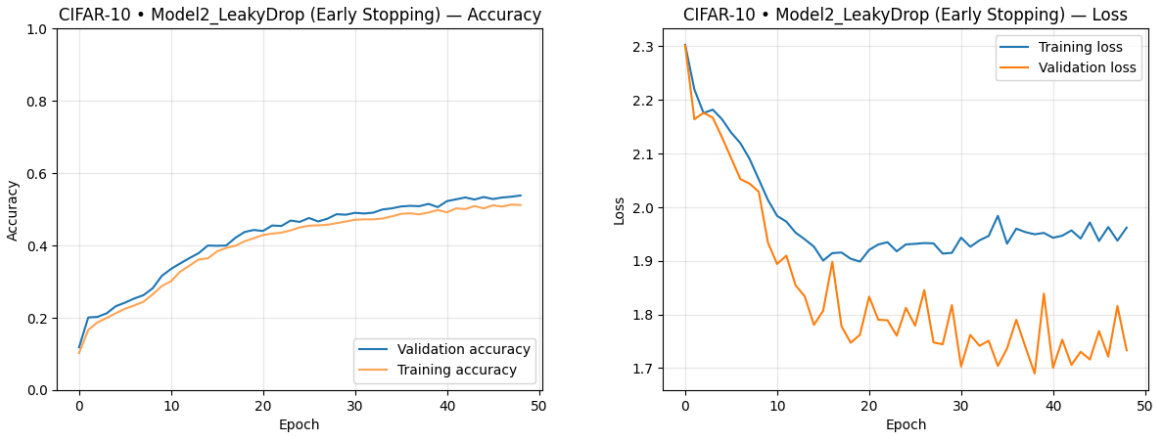


Figure A.20: Accuracy comparison for members and non-members CIFAR-10 DP-No-Regularised with ε = 1.5).

A.1.4 DP-SGD + Regularised (LeakyDropCNN + ES)

LINK: `https://colab.research.google.com/drive/1Sn0-azBkU3skABl1s1785EhkI2MAHE8P?usp=sharing`

Epsilon ε = 0.5



(a) Accuracy. **(b)** Loss.

Figure A.21: Training and validation performance for the CIFAR-10 DP-Regularised with ε = 0.5.

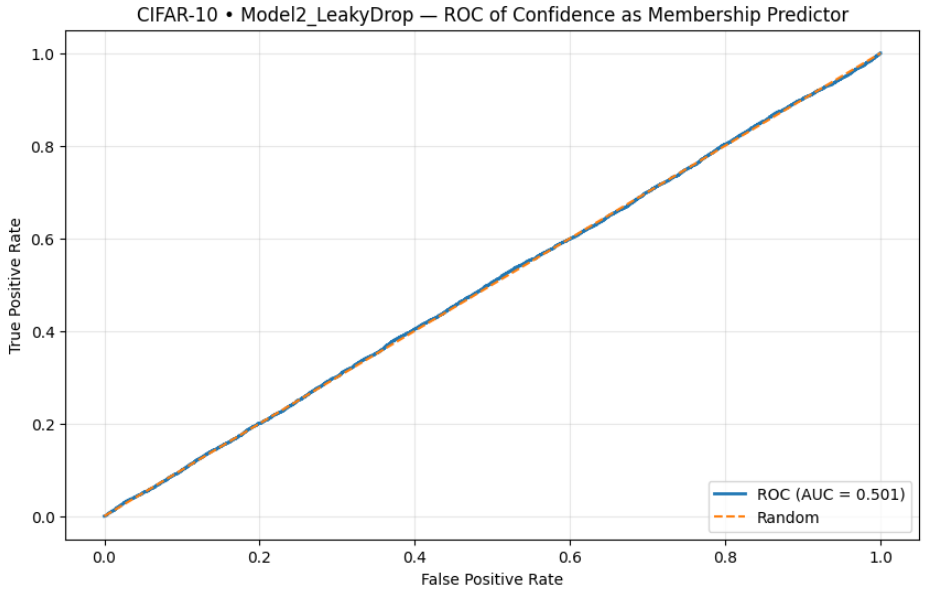


Figure A.22: ROC curve for confidence-based MIA (CIFAR-10 DP-Regularised with ε = 0.5).

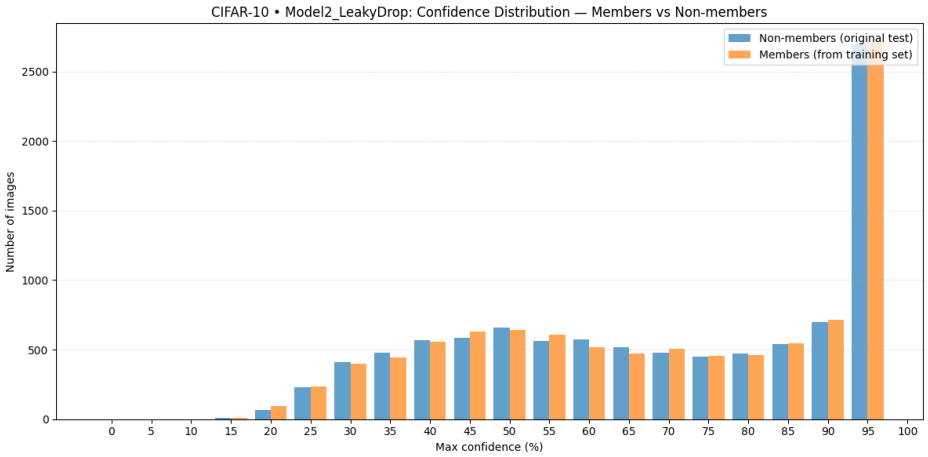


Figure A.23: Confidence distributions for members vs non-members (CIFAR-10 DP-Regularised with $\varepsilon = 0.5$.

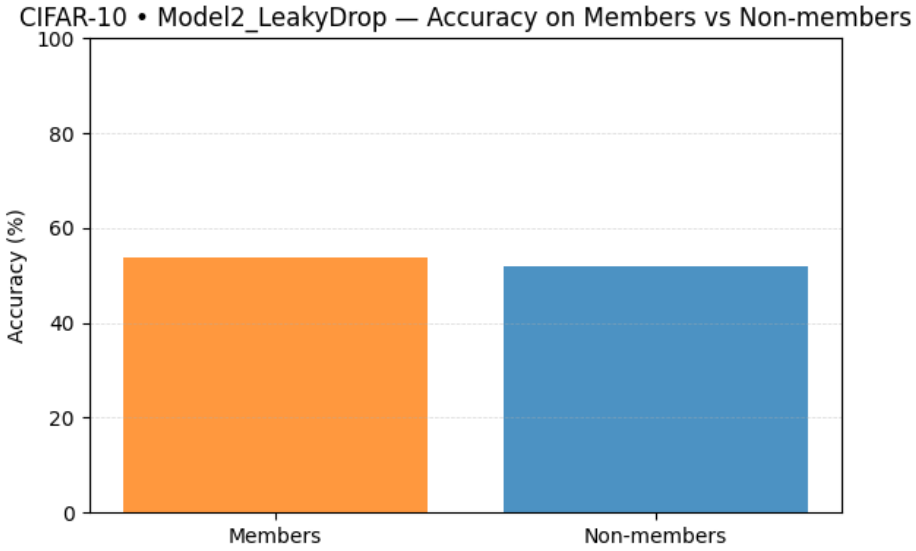
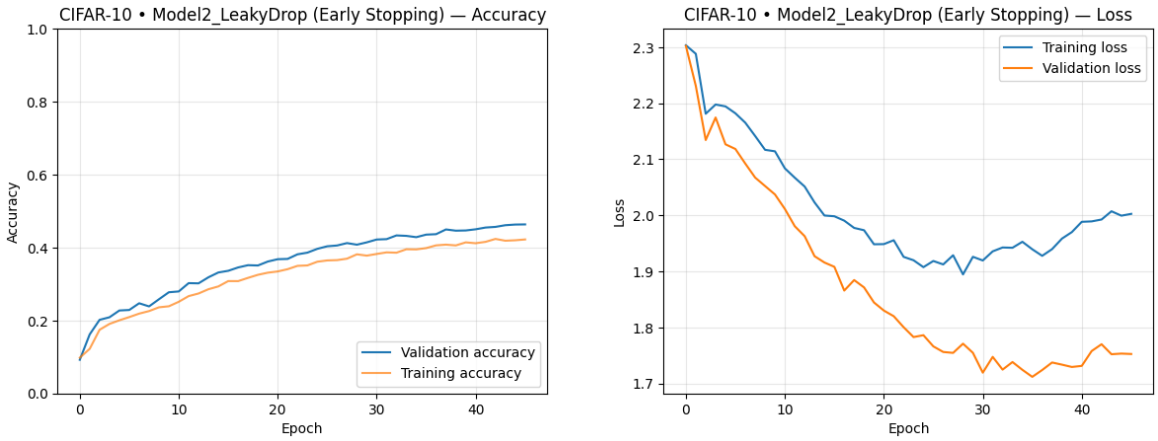


Figure A.24: Accuracy comparison for members and non-members CIFAR-10 DP-Regularised with $\varepsilon = 0.5$).

Epsilon ε = 1.0



(a) Accuracy. **(b)** Loss.

Figure A.25: Training and validation performance for the CIFAR-10 DP-Regularised with ε = 1.0.

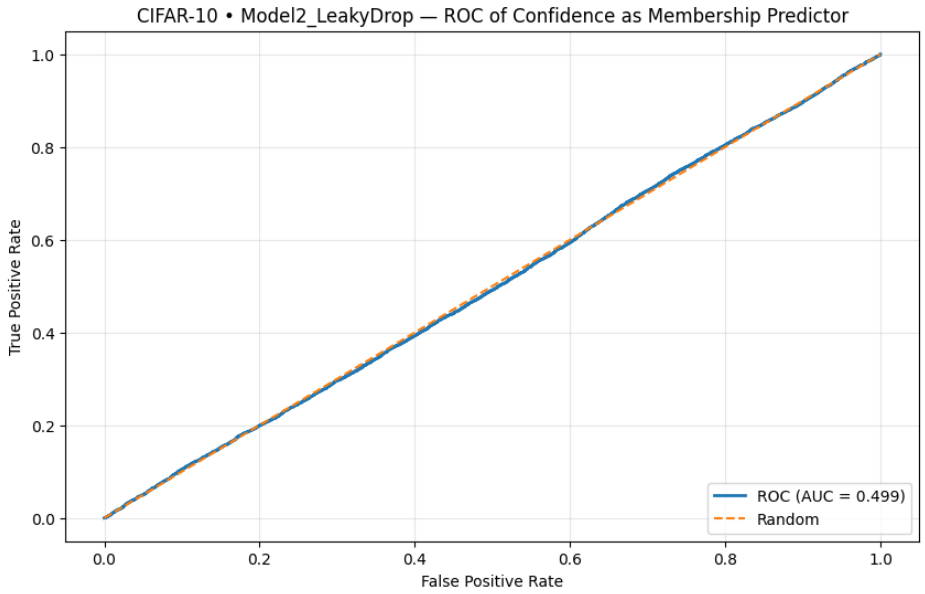


Figure A.26: ROC curve for confidence-based MIA (CIFAR-10 DP-Regularised with ε = 1.0).

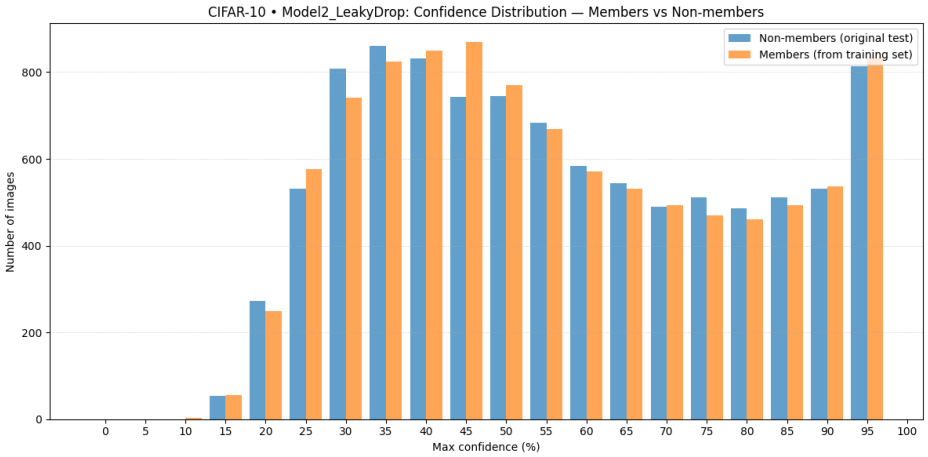


Figure A.27: Confidence distributions for members vs non-members (CIFAR-10 DP-Regularised with $\varepsilon = 1.0$.

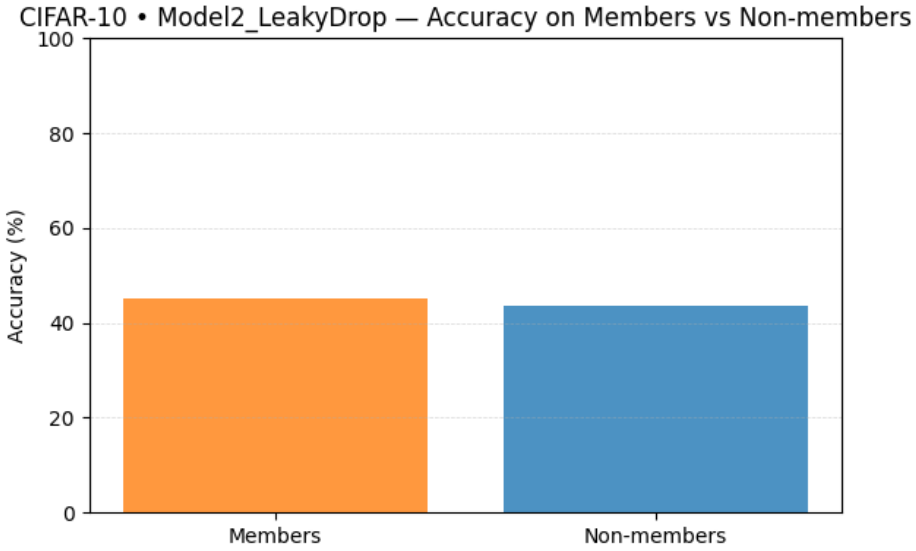
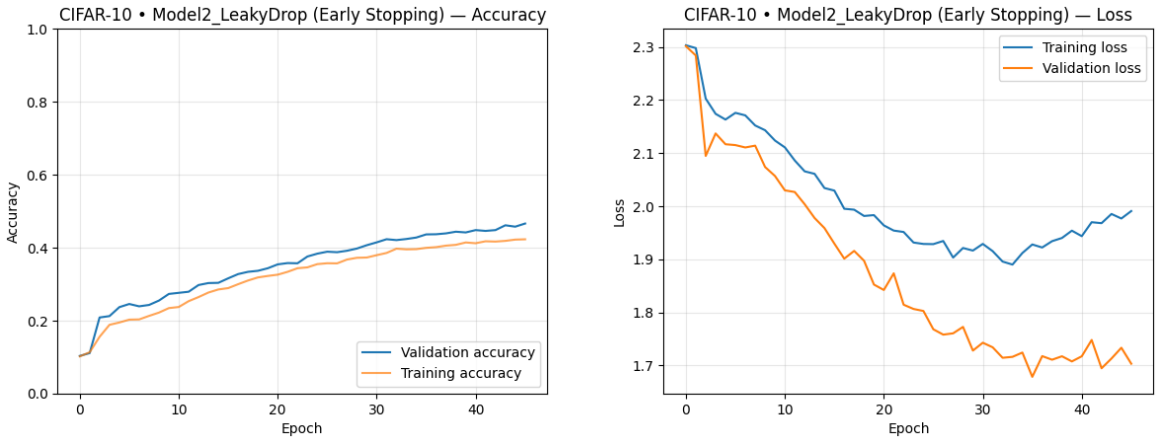


Figure A.28: Accuracy comparison for members and non-members CIFAR-10 DP-Regularised with $\varepsilon = 1.0$).

Epsilon ε = 1.2



(a) Accuracy. **(b)** Loss.

Figure A.29: Training and validation performance for the CIFAR-10 DP-Regularised with ε = 1.2.

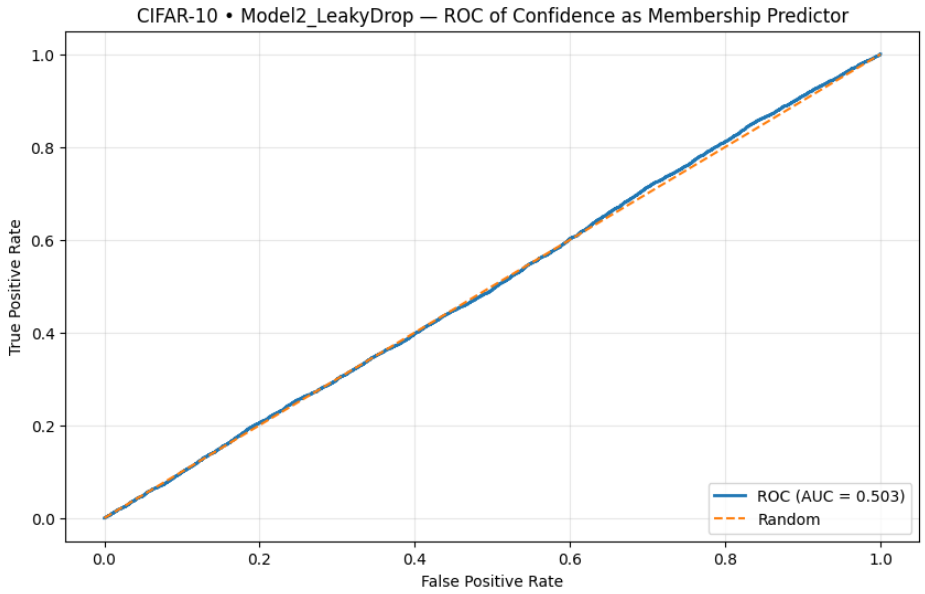


Figure A.30: ROC curve for confidence-based MIA (CIFAR-10 DP-Regularised with ε = 1.2).

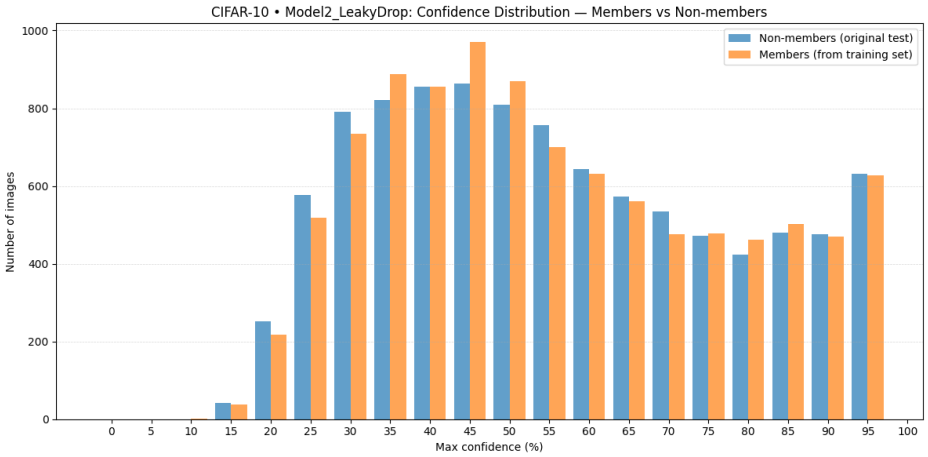


Figure A.31: Confidence distributions for members vs non-members (CIFAR-10 DP-Regularised with $\varepsilon = 1.2$.

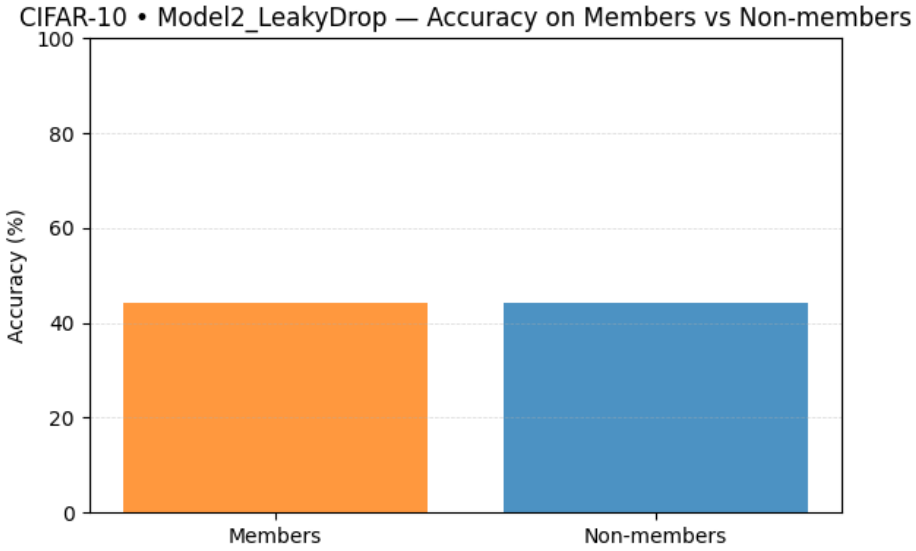
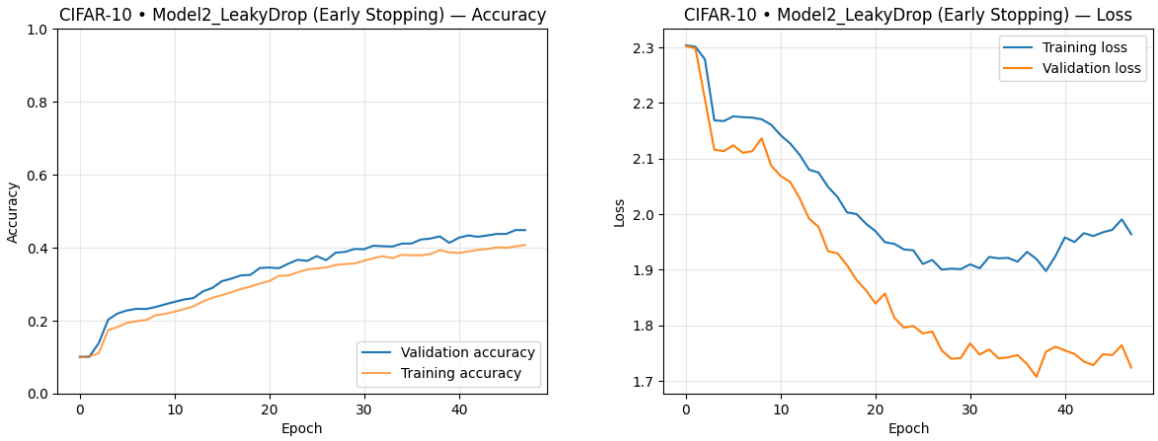


Figure A.32: Accuracy comparison for members and non-members CIFAR-10 DP-Regularised with $\varepsilon = 1.2$).

Epsilon ε = 1.5



(a) Accuracy. **(b)** Loss.

Figure A.33: Training and validation performance for the CIFAR-10 DP-Regularised with ε = 1.5.

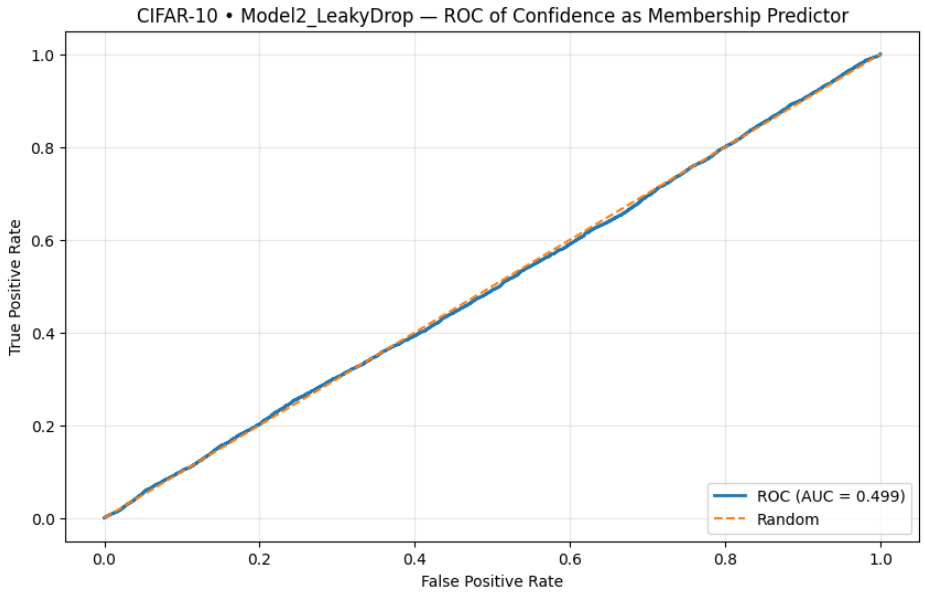


Figure A.34: ROC curve for confidence-based MIA (CIFAR-10 DP-Regularised with ε = 1.5).

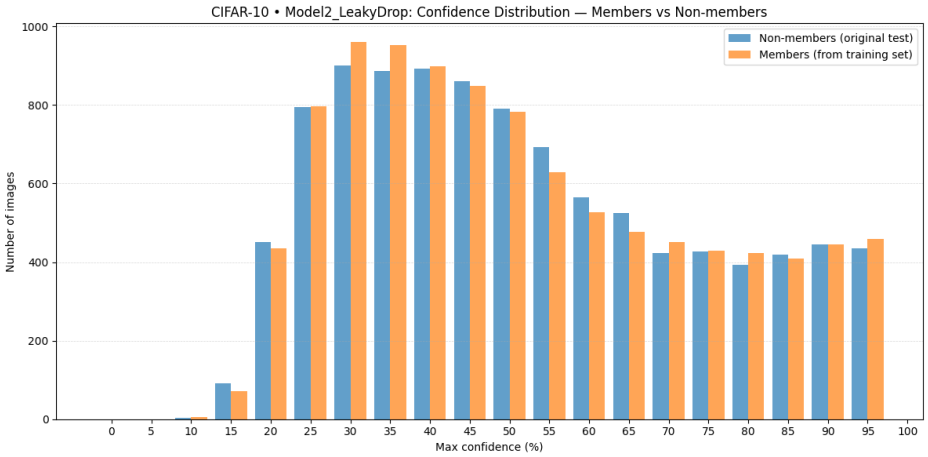


Figure A.35: Confidence distributions for members vs non-members (CIFAR-10 DP-Regularised with $\varepsilon = 1.5$.

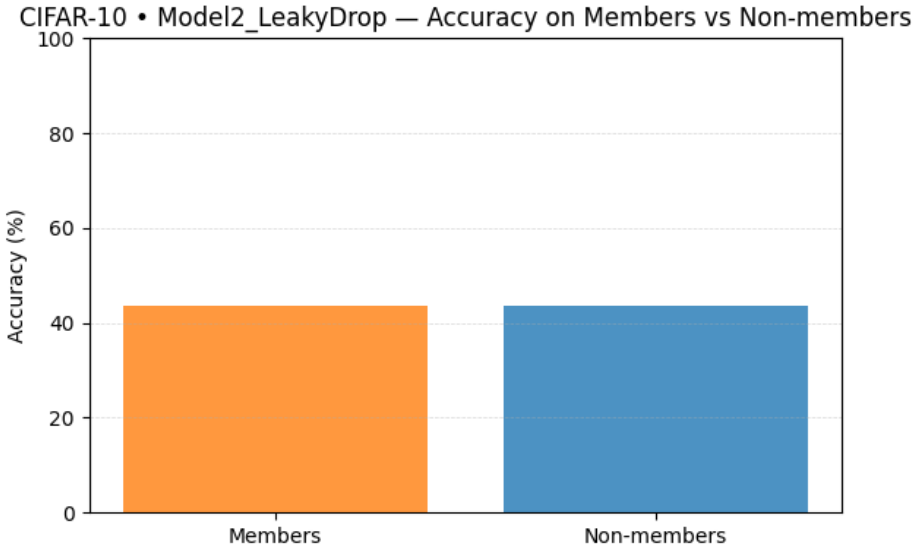


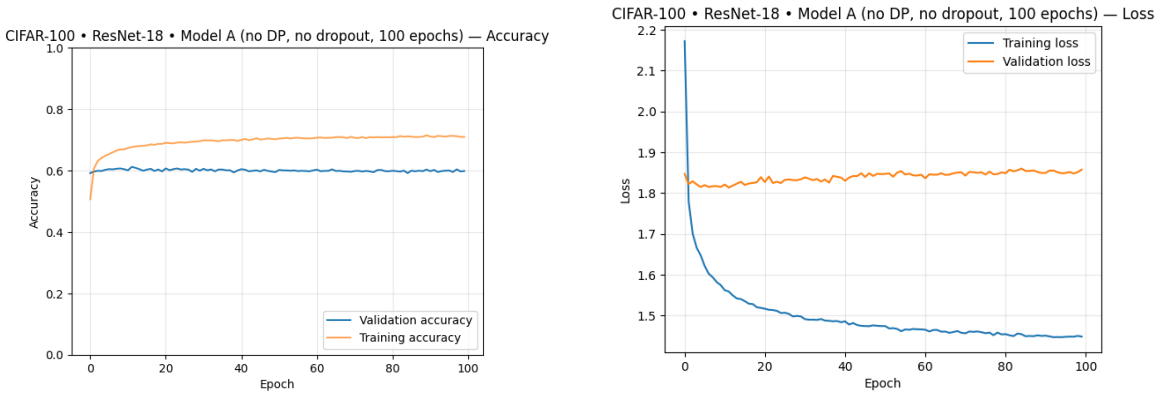
Figure A.36: Accuracy comparison for members and non-members CIFAR-10 DP-Regularised with $\varepsilon = 1.5$).

A.2 Cifar-100

A.2.1 ResNet18 /w Transfer

Baseline

Link: `https://colab.research.google.com/drive/1NcocoSXUSFIofDL0VACJVhzbSUy_Q243?usp=sharing`



(a) Accuracy. **(b)** Loss.

Figure A.37: Training and validation performance for the CIFAR-100 /w Transfer-Baseline model.

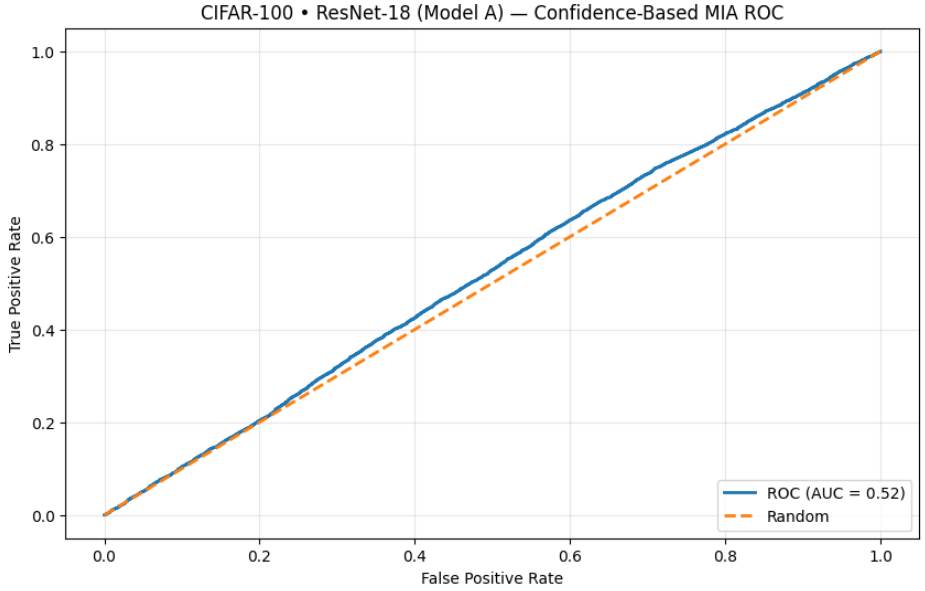


Figure A.38: ROC curve for confidence-based MIA (CIFAR-100 /w Transfer-Baseline model).

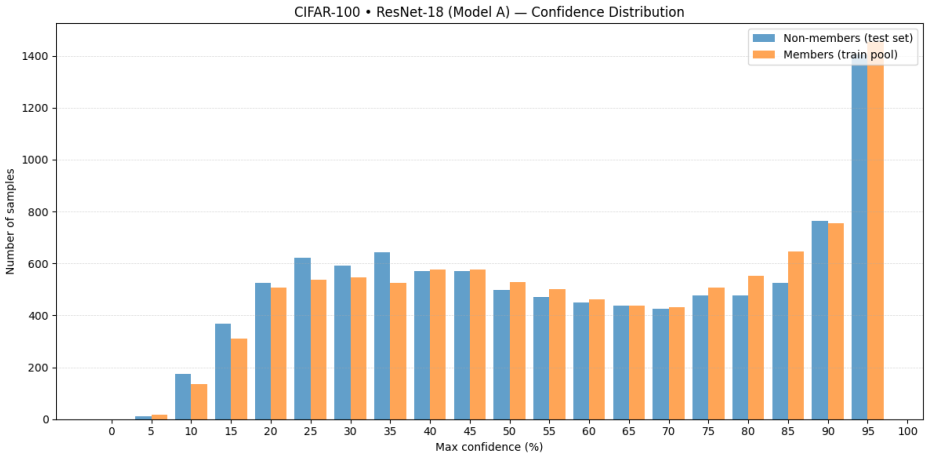


Figure A.39: Confidence distributions for members vs non-members (CIFAR-100 /w Transfer-Baseline model.

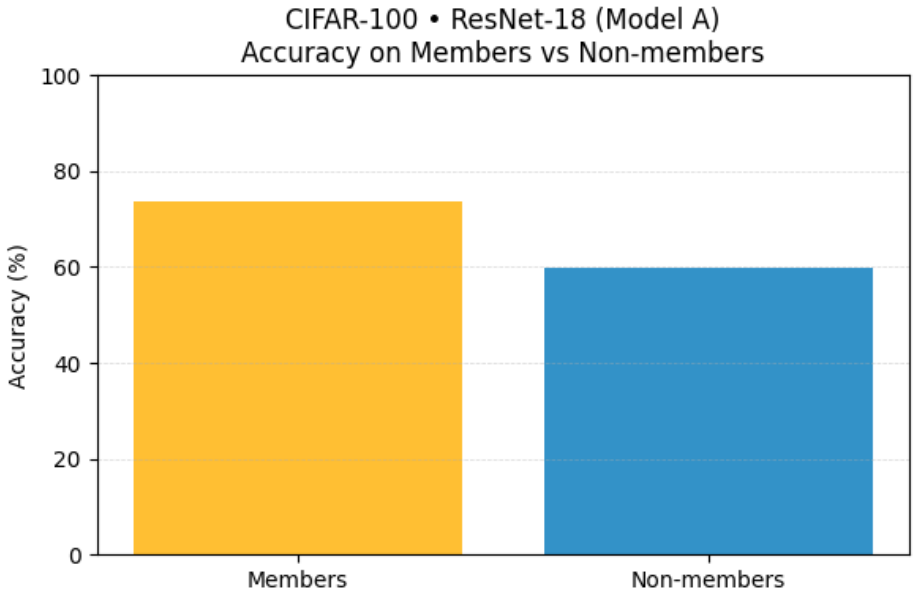
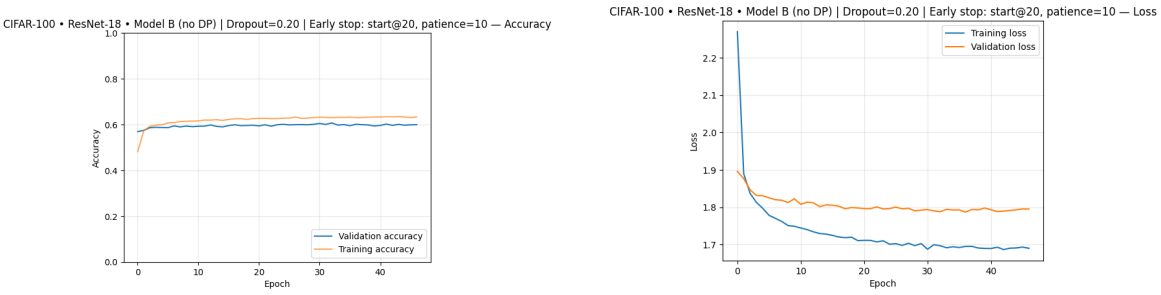


Figure A.40: Accuracy comparison for members and non-members (CIFAR-100 /w Transfer-Baseline model).

Regularized

LINK: `https://colab.research.google.com/drive/1XO5mI3MX_qc08KmQrvBW7sBPGDhUuJI3?usp=sharing`



(a) Accuracy. **(b)** Loss.

Figure A.41: Training and validation performance for the CIFAR-100 /w Transfer-Regularized model.

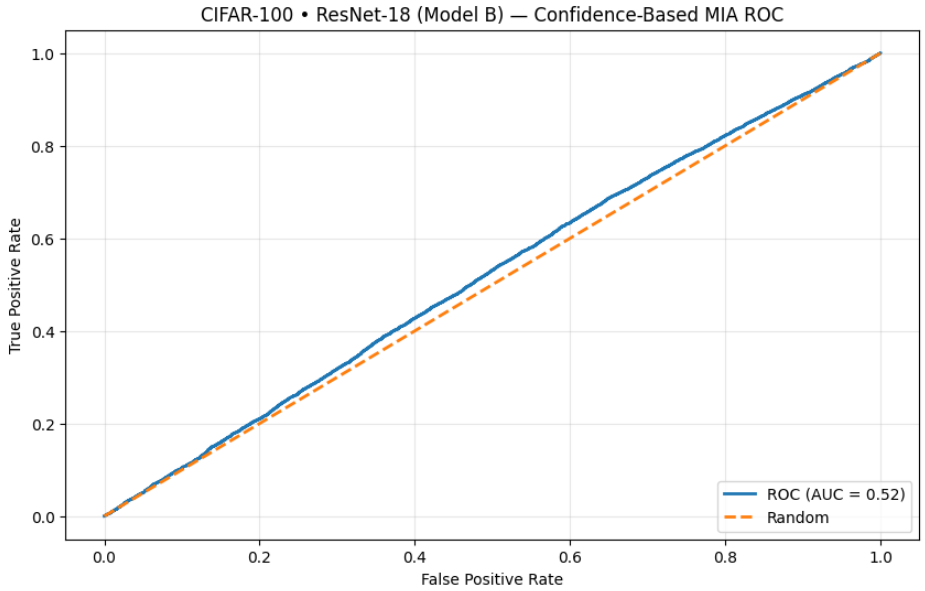


Figure A.42: ROC curve for confidence-based MIA (CIFAR-100 /w Transfer-Regularized model).

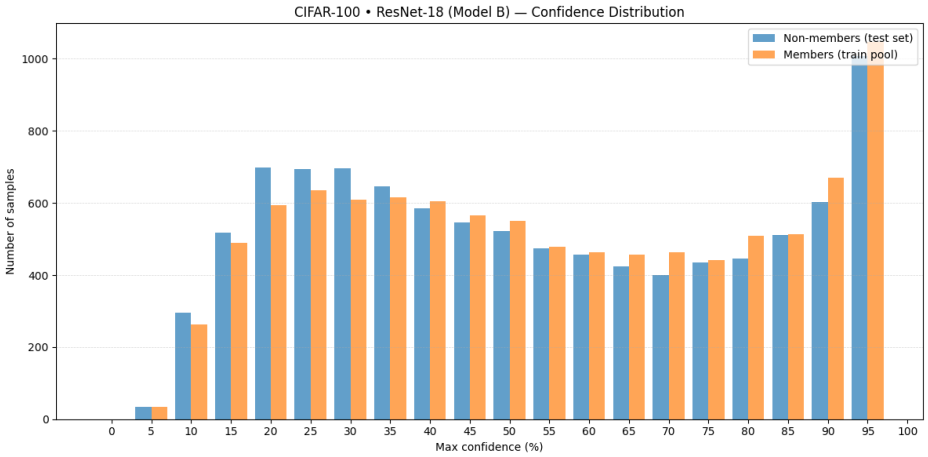


Figure A.43: Confidence distributions for members vs non-members (CIFAR-100 /w Transfer-Regularized model.

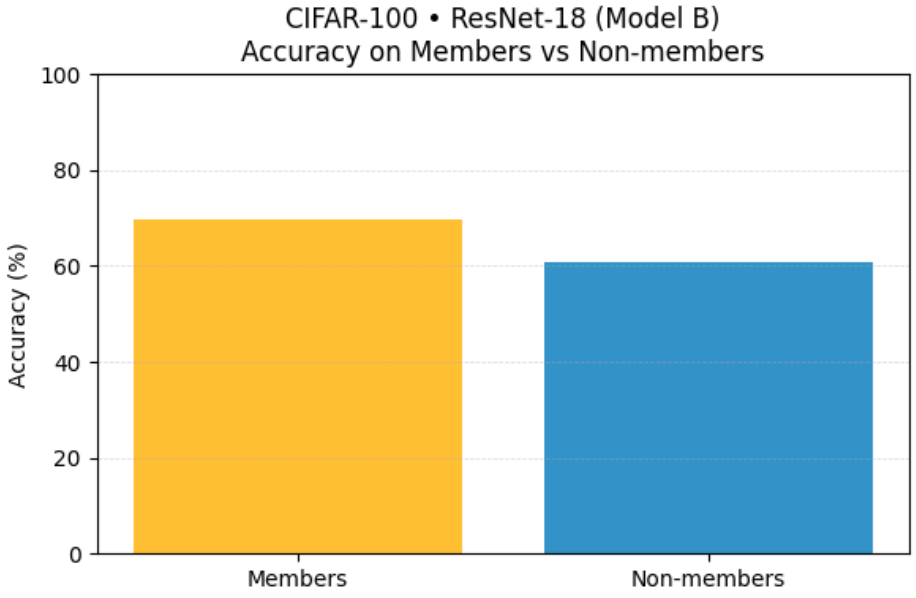
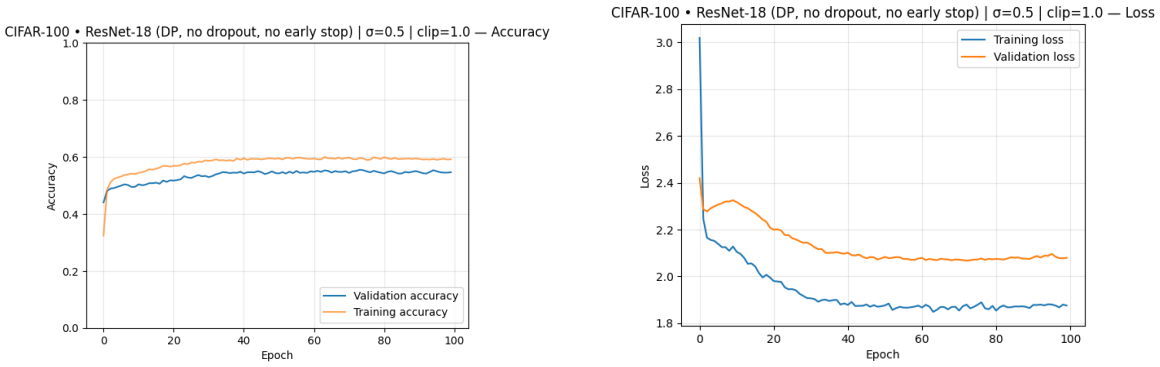


Figure A.44: Accuracy comparison for members and non-members (CIFAR-100 /w Transfer-Regularized model).

DP no Regularisation

Epsilon $\varepsilon = 0.5$ LINK: `https://colab.research.google.com/drive/1QM-Ke5xZFgCf1jc4x31tpkW7_mY4AbY_?usp=sharing`



(a) Accuracy. **(b)** Loss.

Figure A.45: Training and validation performance for the CIFAR-100 /w Transfer + DP +Epsilon $\varepsilon = 0.5$.

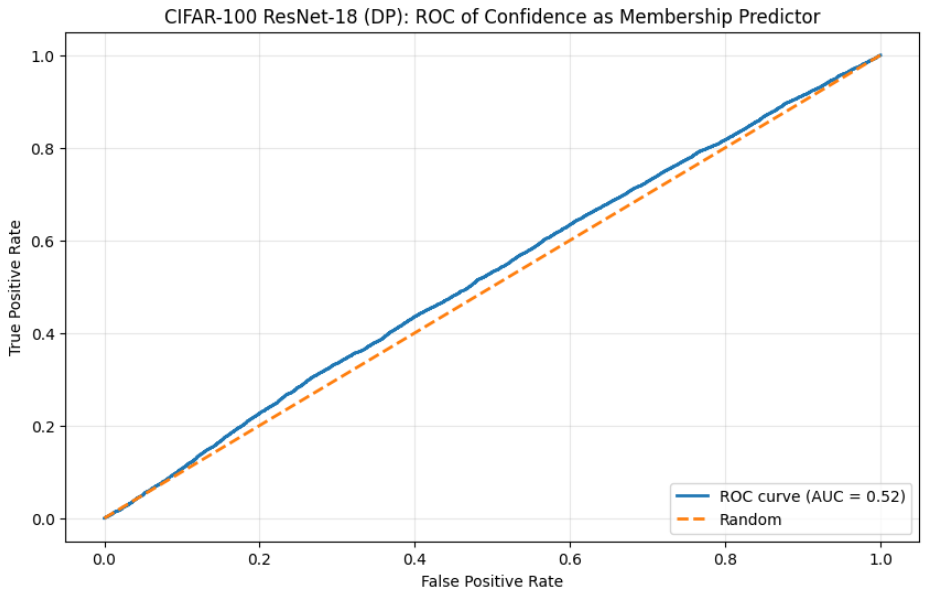


Figure A.46: ROC curve for confidence-based MIA (CIFAR-100 /w Transfer + DP + Epsilon $\varepsilon = 0.5$).

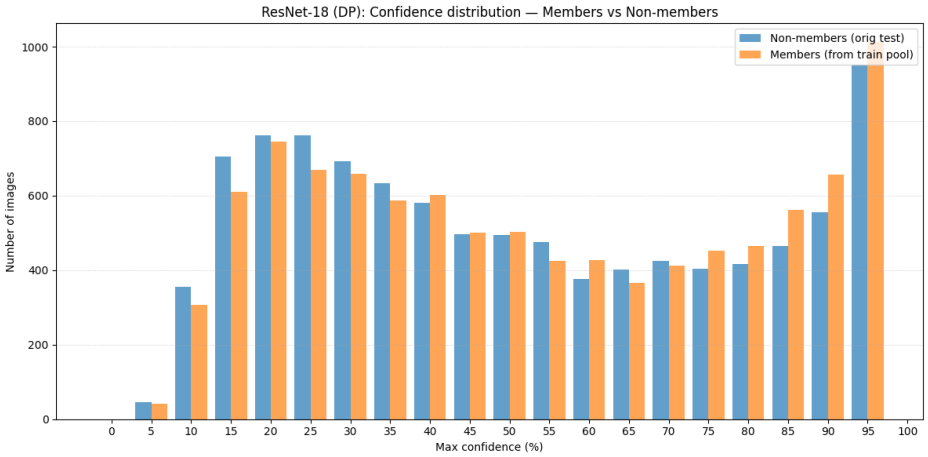


Figure A.47: Confidence distributions for members vs non-members (CIFAR-100 + DP + Epsilon ε = 0.5.

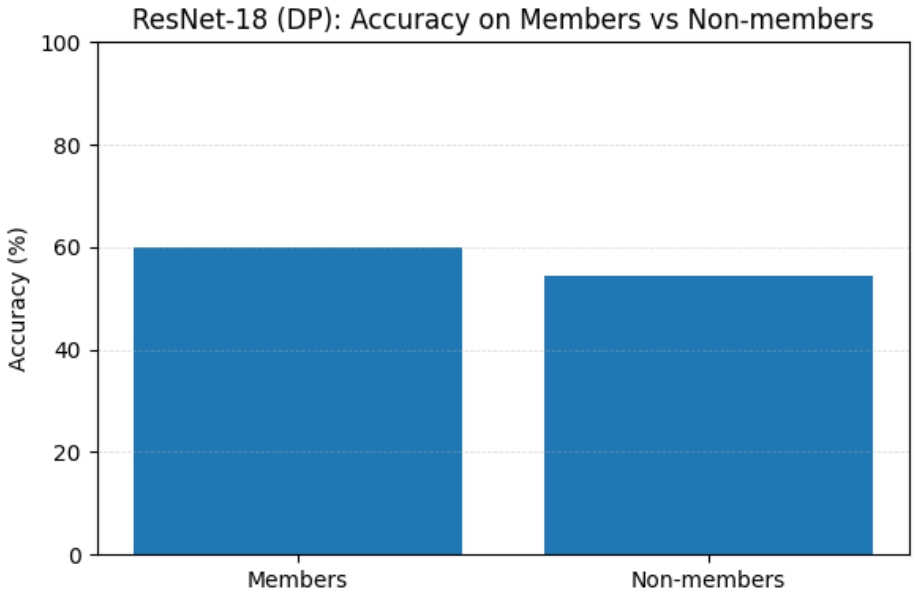
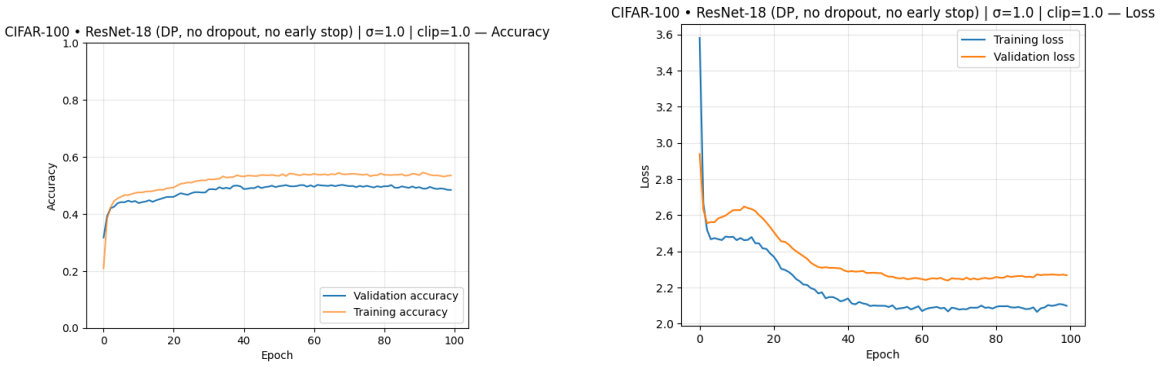


Figure A.48: Accuracy comparison for members and non-members (CIFAR-100 + DP + Epsilon ε = 0.5).

Epsilon ε = 1.0 LINK: `https://colab.research.google.com/drive/17VsQoEb-o2irZ2jogvtPQ4KoI0-srkg?usp=sharing`



(a) Accuracy. **(b)** Loss.

Figure A.49: Training and validation performance for the CIFAR-100 /w Transfer + DP +Epsilon ε = 1.0.

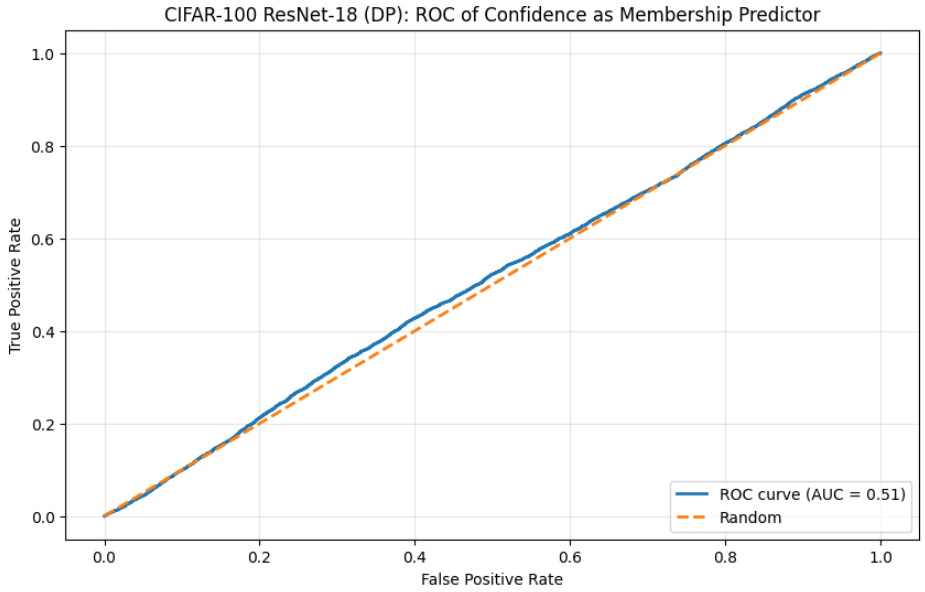


Figure A.50: ROC curve for confidence-based MIA (CIFAR-100 /w Transfer + DP + Epsilon ε = 1.0).

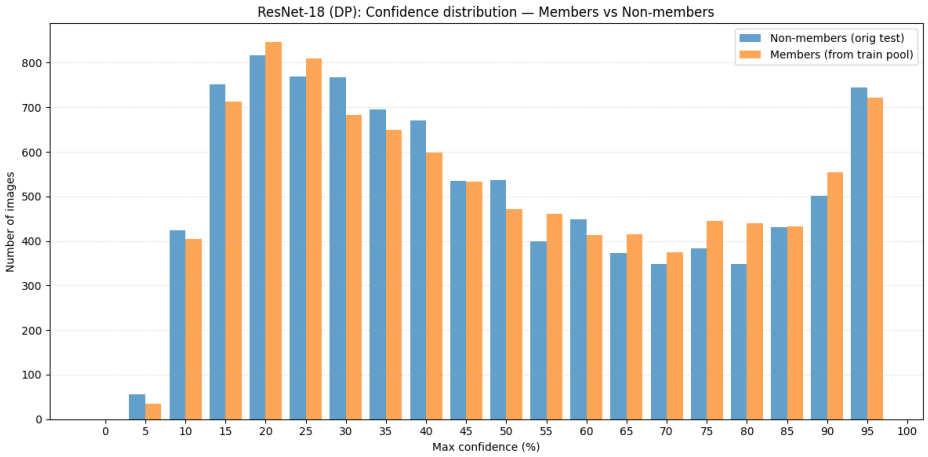


Figure A.51: Confidence distributions for members vs non-members (CIFAR-100 + DP + Epsilon ε = 1.0.

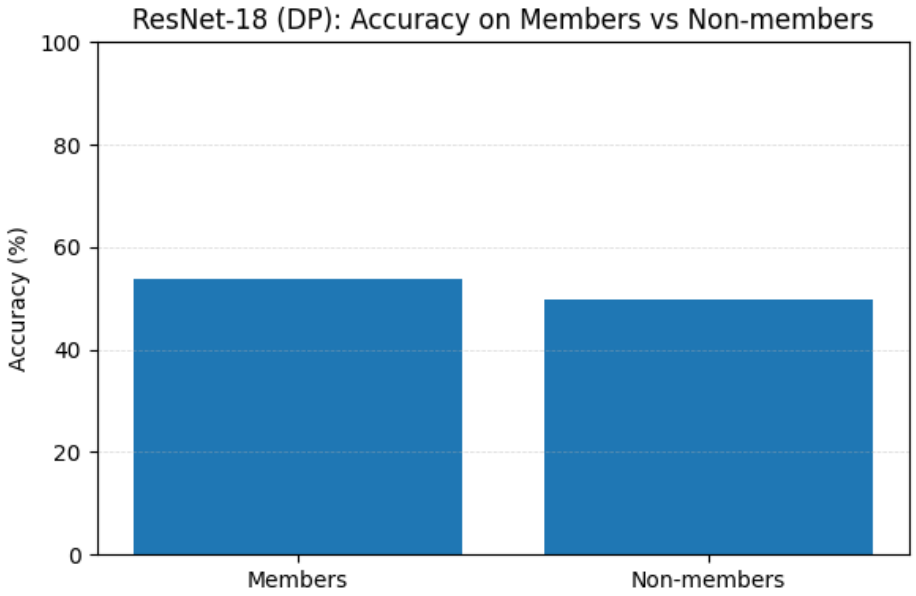
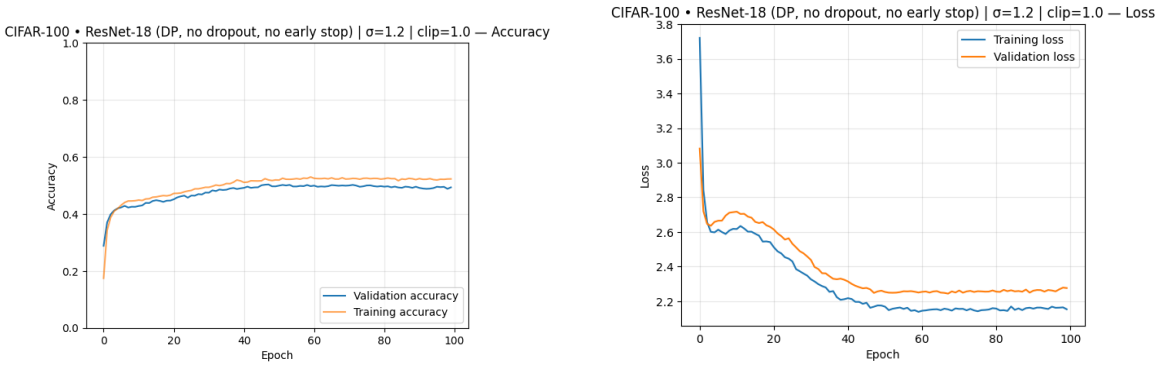


Figure A.52: Accuracy comparison for members and non-members (CIFAR-100 + DP + Epsilon ε = 1.0).

Epsilon ε = 1.2 LINK: `https://colab.research.google.com/drive/16HpmOvyDafER2_KUeLPKITPTeMW3VAr`
`usp=sharing`



(a) Accuracy. **(b)** Loss.

Figure A.53: Training and validation performance for the CIFAR-100 /w Transfer + DP +Epsilon ε = 1.2.

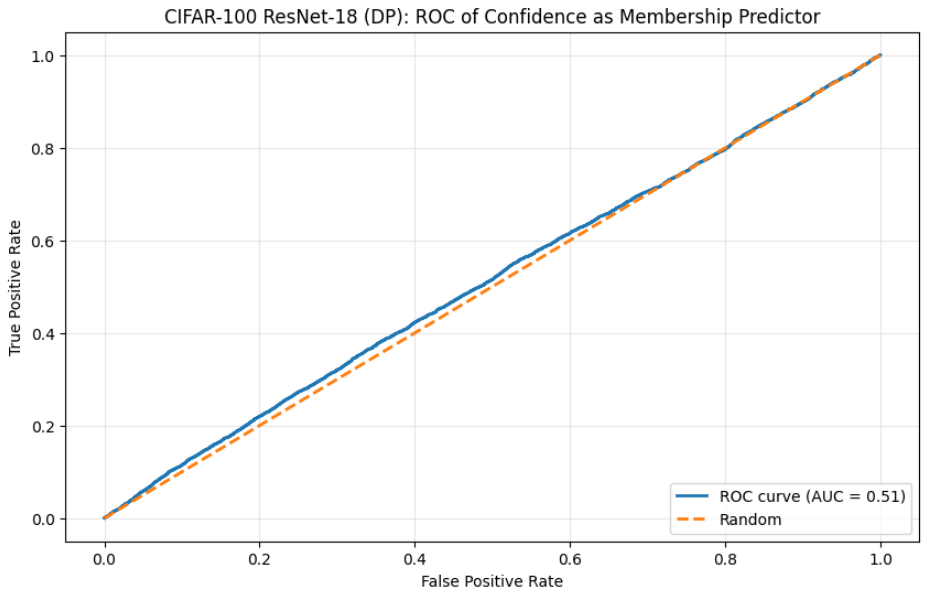


Figure A.54: ROC curve for confidence-based MIA (CIFAR-100 /w Transfer + DP + Epsilon ε = 1.2).

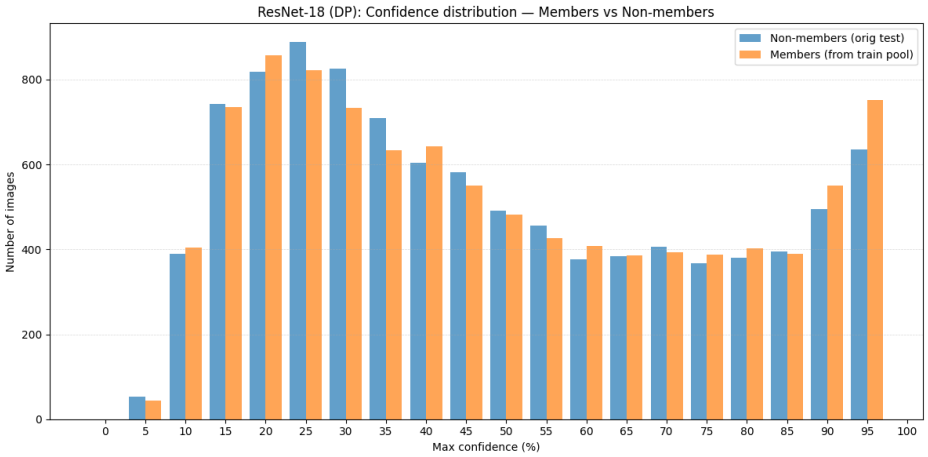


Figure A.55: Confidence distributions for members vs non-members (CIFAR-100 + DP + Epsilon ε = 1.2.

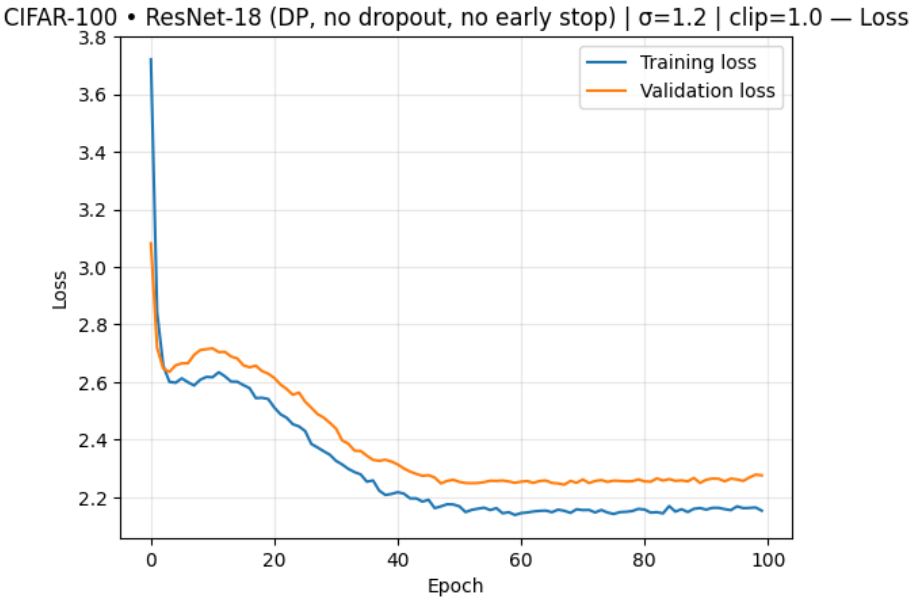
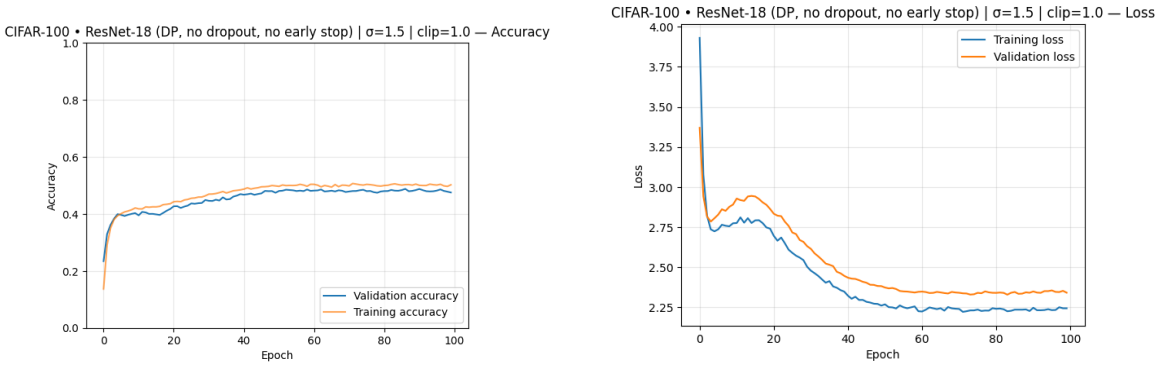


Figure A.56: Accuracy comparison for members and non-members (CIFAR-100 + DP + Epsilon ε = 1.2).

Epsilon ε = 1.5 LINK: `https://colab.research.google.com/drive/1npHt02No5aMMTRhKDsN8O1QW8OUJ49E?usp=sharing`



(a) Accuracy. **(b)** Loss.

Figure A.57: Training and validation performance for the CIFAR-100 /w Transfer + DP +Epsilon $\varepsilon = 1.5$.

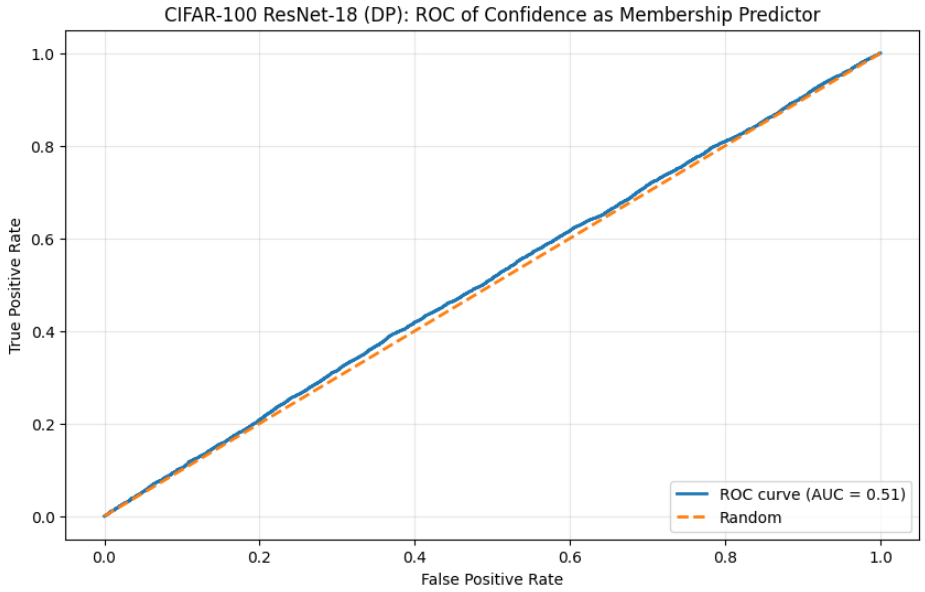


Figure A.58: ROC curve for confidence-based MIA (CIFAR-100 /w Transfer + DP + Epsilon $\varepsilon = 1.5$).

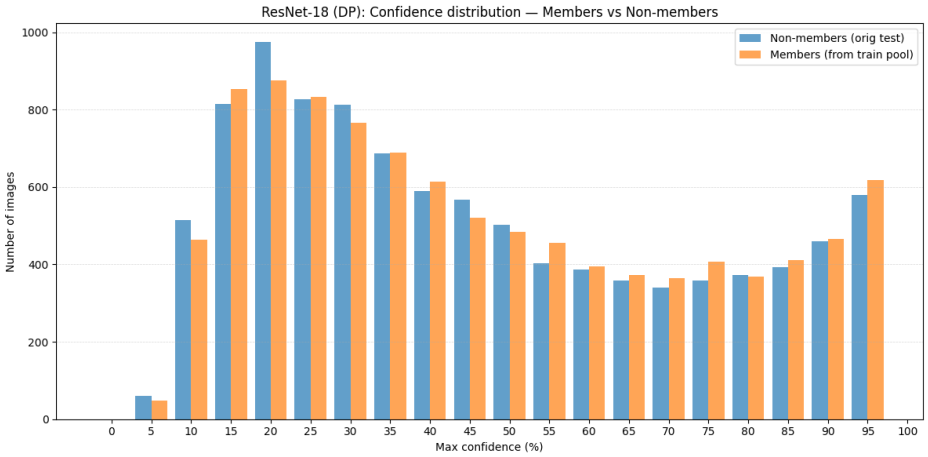


Figure A.59: Confidence distributions for members vs non-members (CIFAR-100 + DP + Epsilon ε = 1.5.

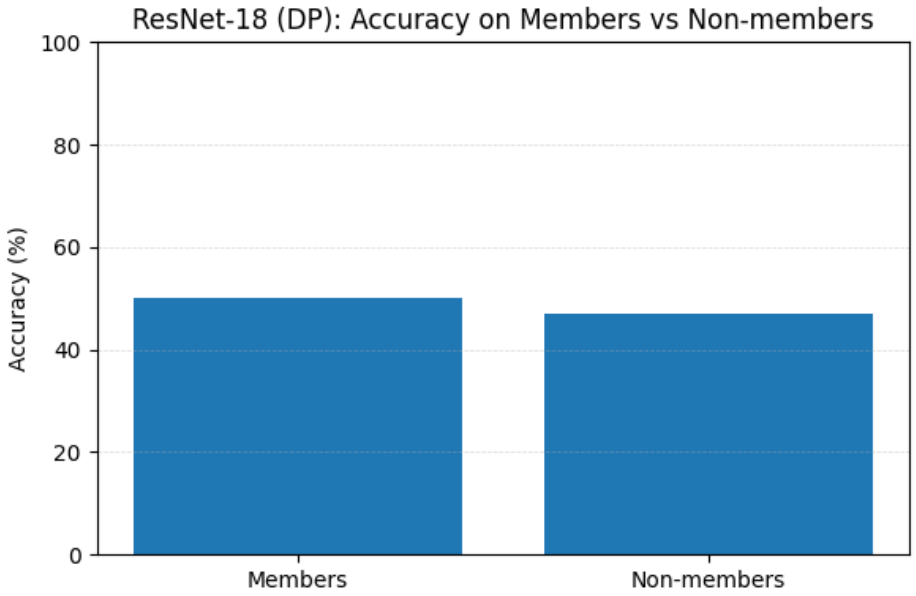
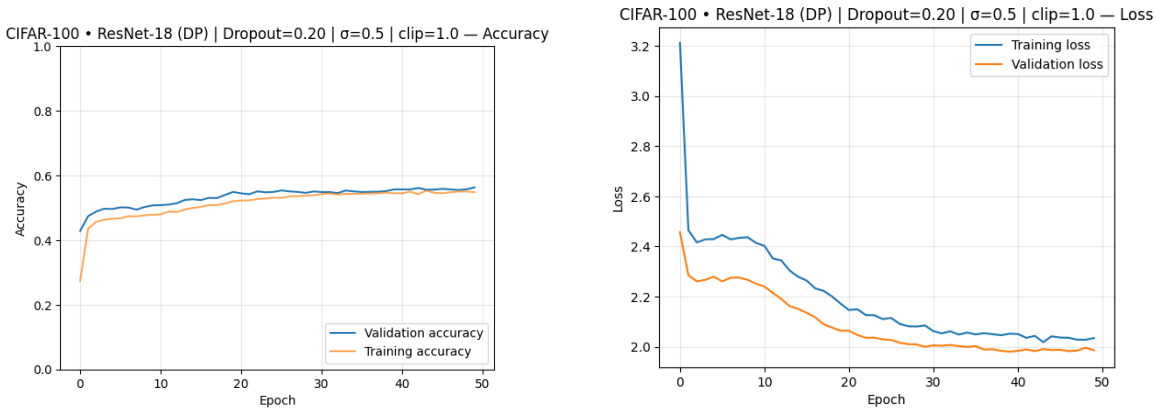


Figure A.60: Accuracy comparison for members and non-members (CIFAR-100 + DP + Epsilon ε = 1.5).

DP + Regularisation

Epsilon ε = 0.5 LINK: `https://colab.research.google.com/drive/1fzOmfPy3QiRaq5MMxKWWQc-TnvlFyEf usp=sharing`



(a) Accuracy.

(b) Loss.

Figure A.61: Training and validation performance for the CIFAR-100 /w Transfer-Regularized model + DP +Epsilon ε = 0.5.

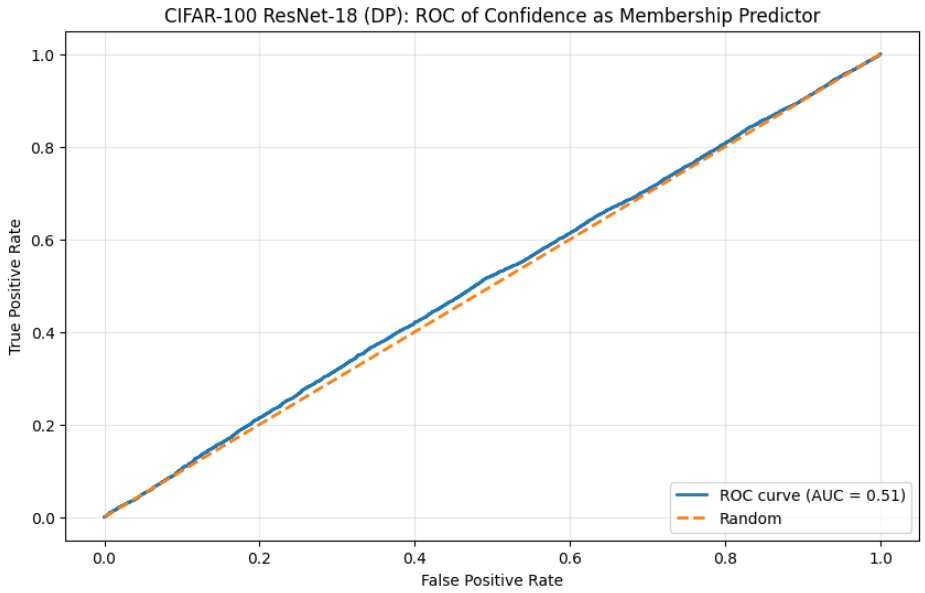


Figure A.62: ROC curve for confidence-based MIA (CIFAR-100 /w Transfer-Regularized model + DP + Epsilon ε = 0.5).

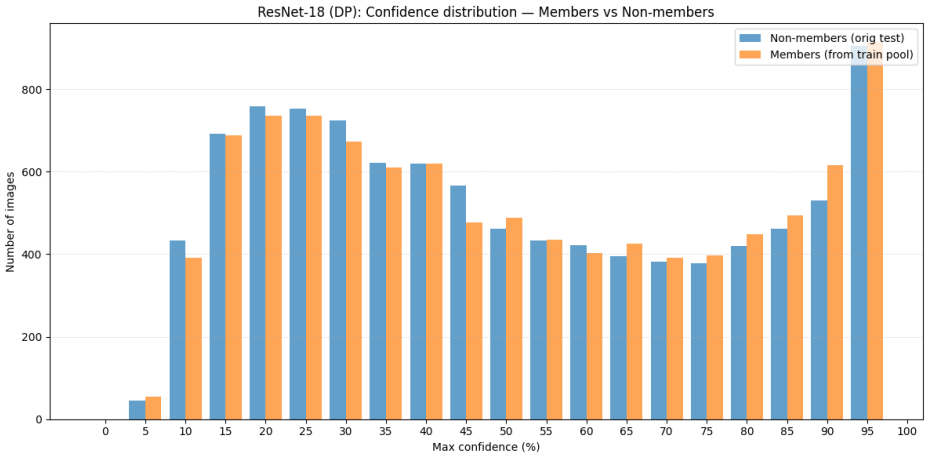


Figure A.63: Confidence distributions for members vs non-members (CIFAR-100 /w Transfer-Regularized model + DP + Epsilon $\varepsilon = 0.5$.

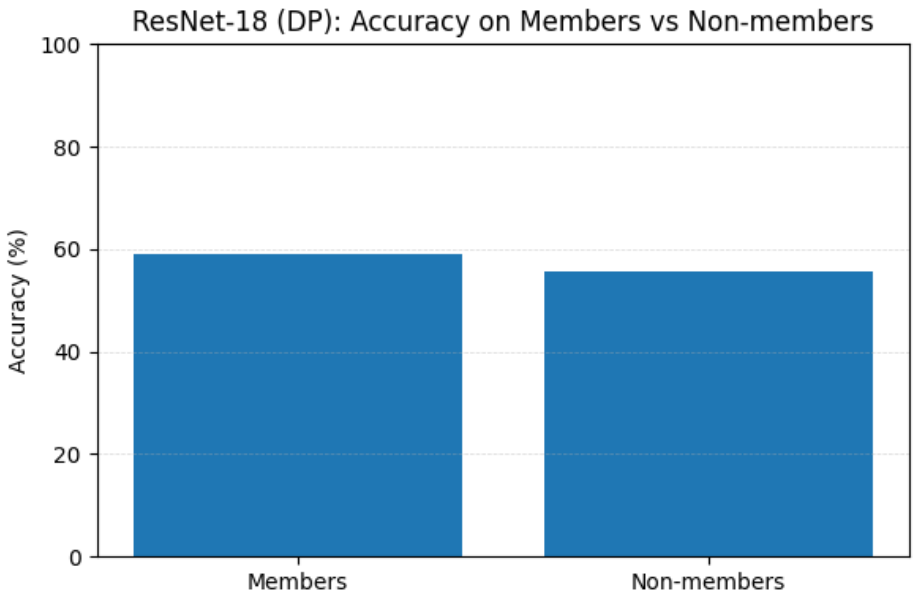
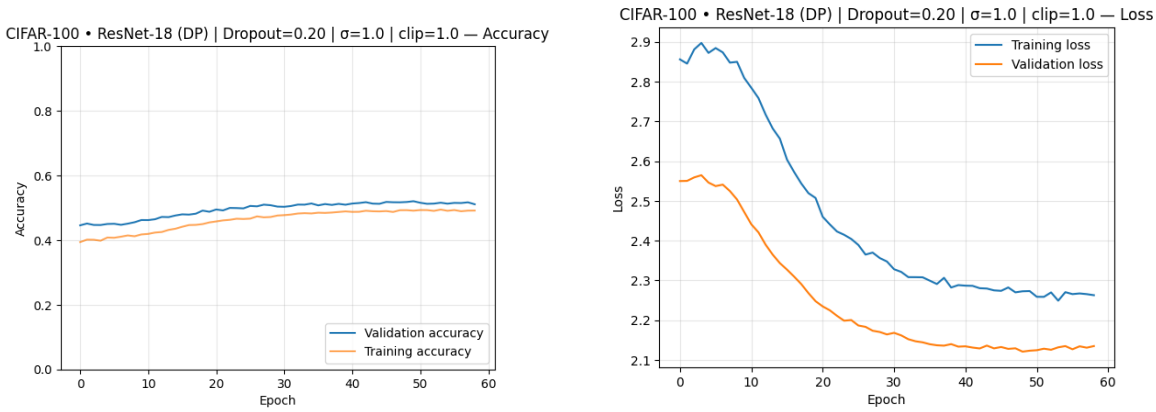


Figure A.64: Accuracy comparison for members and non-members (CIFAR-100 /w Transfer-Regularized model + DP + Epsilon $\varepsilon = 0.5$).

Epsilon $\varepsilon = 1.0$ LINK: `https://colab.research.google.com/drive/1AlNTr4ZtG3TTJE8HdXOF2LfGk65K_W3Z?usp=sharing`



(a) Accuracy. **(b)** Loss.

Figure A.65: Training and validation performance for the CIFAR-100 /w Transfer-Regularized model + DP + Epsilon ε = 1.0.

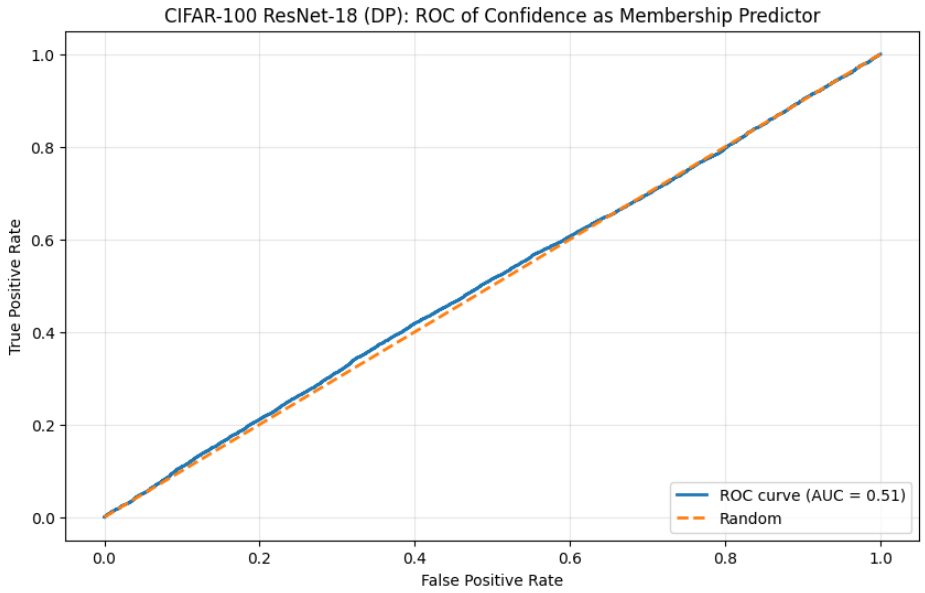


Figure A.66: ROC curve for confidence-based MIA (CIFAR-100 /w Transfer-Regularized model + DP + Epsilon ε = 1.0).

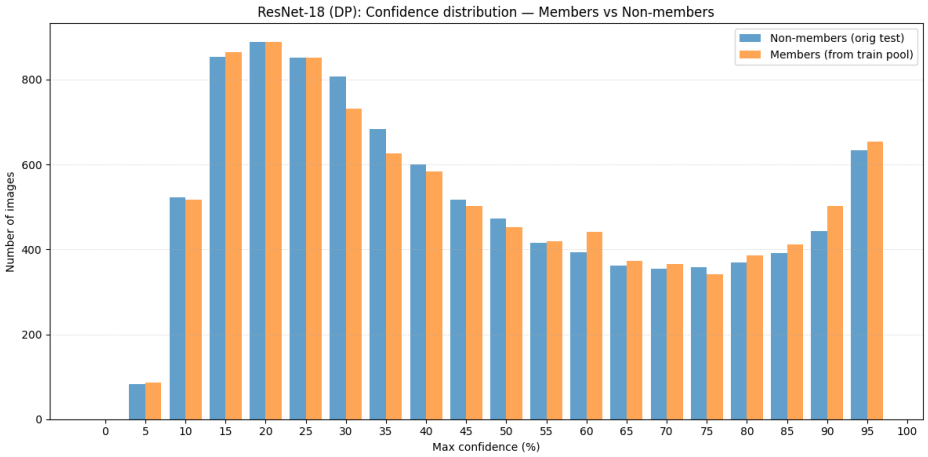


Figure A.67: Confidence distributions for members vs non-members (CIFAR-100 /w Transfer-Regularized model + DP + Epsilon ε = 1.0.

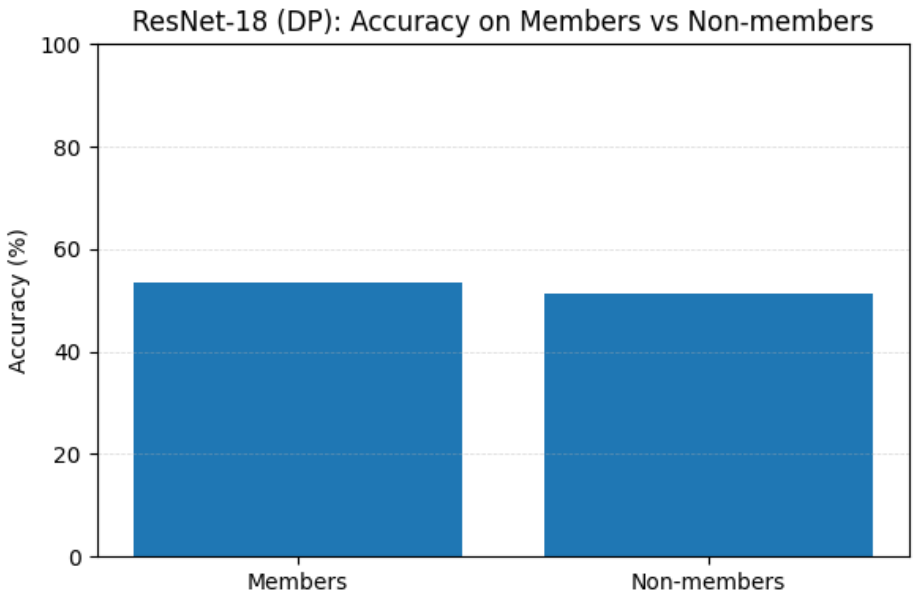
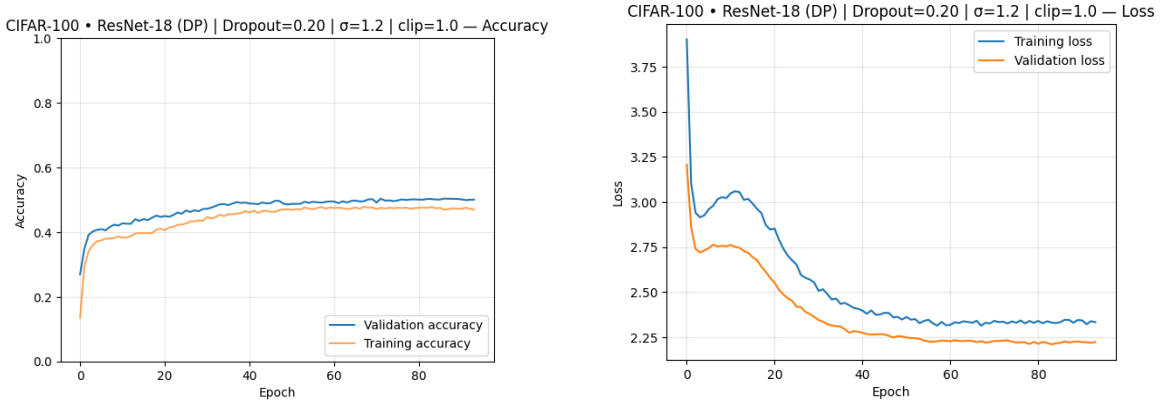


Figure A.68: Accuracy comparison for members and non-members (CIFAR-100 /w Transfer-Regularized model + DP + Epsilon ε = 1.0).

Epsilon ε = 1.2 LINK: `https://colab.research.google.com/drive/1OoUvML3dpmvlh-uV6RazXhlyN2AYtt8usp=sharing`



(a) Accuracy. **(b)** Loss.

Figure A.69: Training and validation performance for the CIFAR-100 /w Transfer-Regularized model + DP + Epsilon $\varepsilon = 1.2$.

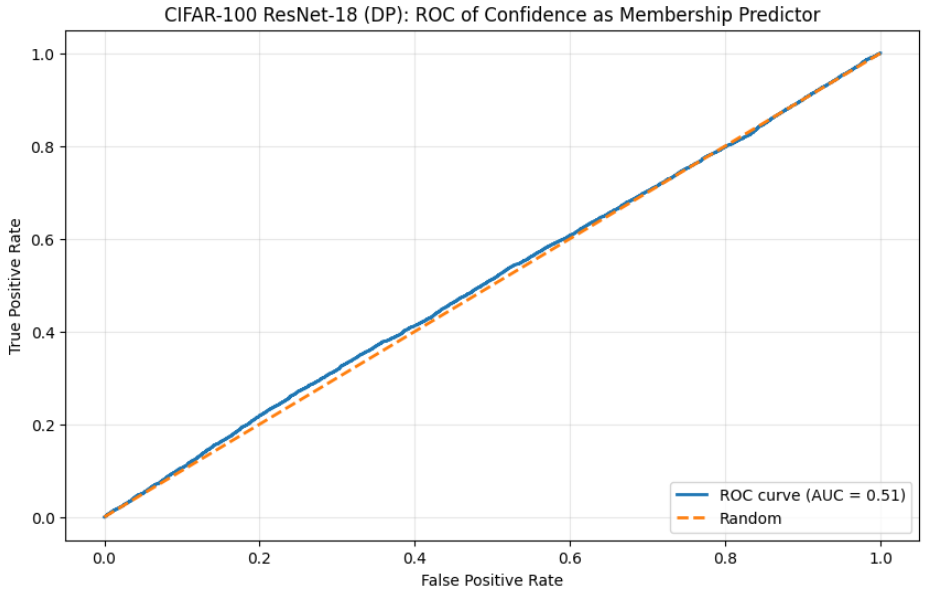


Figure A.70: ROC curve for confidence-based MIA (CIFAR-100 /w Transfer-Regularized model + DP + Epsilon $\varepsilon = 1.2$).

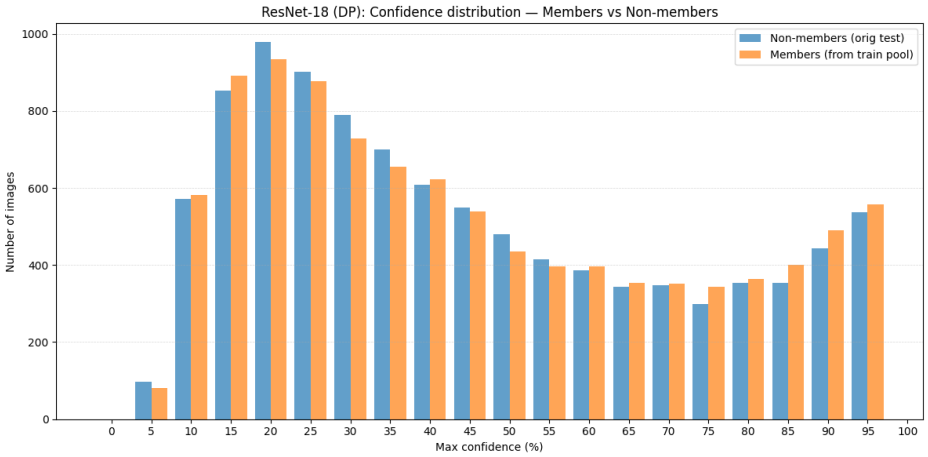


Figure A.71: Confidence distributions for members vs non-members (CIFAR-100 /w Transfer-Regularized model + DP + Epsilon ε = 1.2.

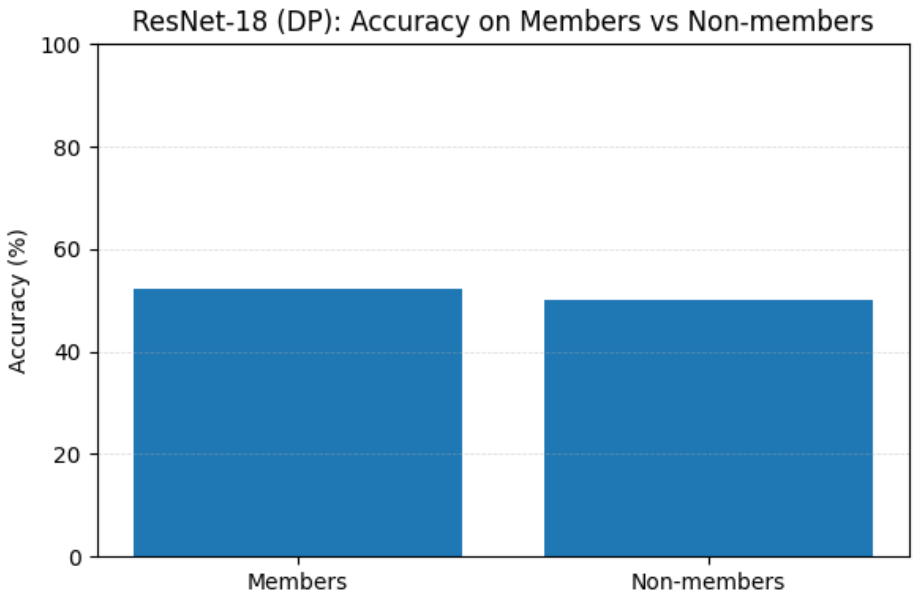
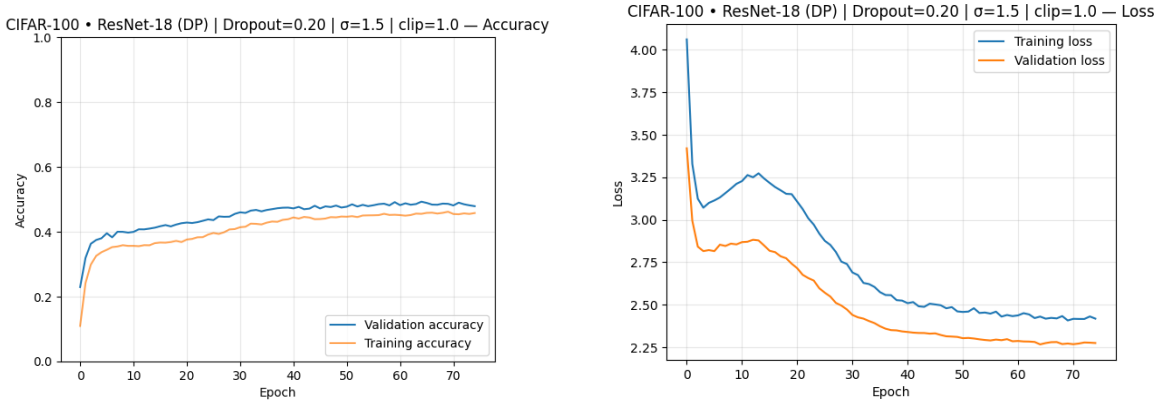


Figure A.72: Accuracy comparison for members and non-members (CIFAR-100 /w Transfer-Regularized model + DP + Epsilon ε = 1.2).

Epsilon ε = 1.5 LINK: `https://colab.research.google.com/drive/171X8hRf0tiHLJ31NHB-UfTABwc4SOc0usp=sharing`



(a) Accuracy.

(b) Loss.

Figure A.73: Training and validation performance for the CIFAR-100 /w Transfer-Regularized model + DP + Epsilon $\varepsilon = 1.5$.

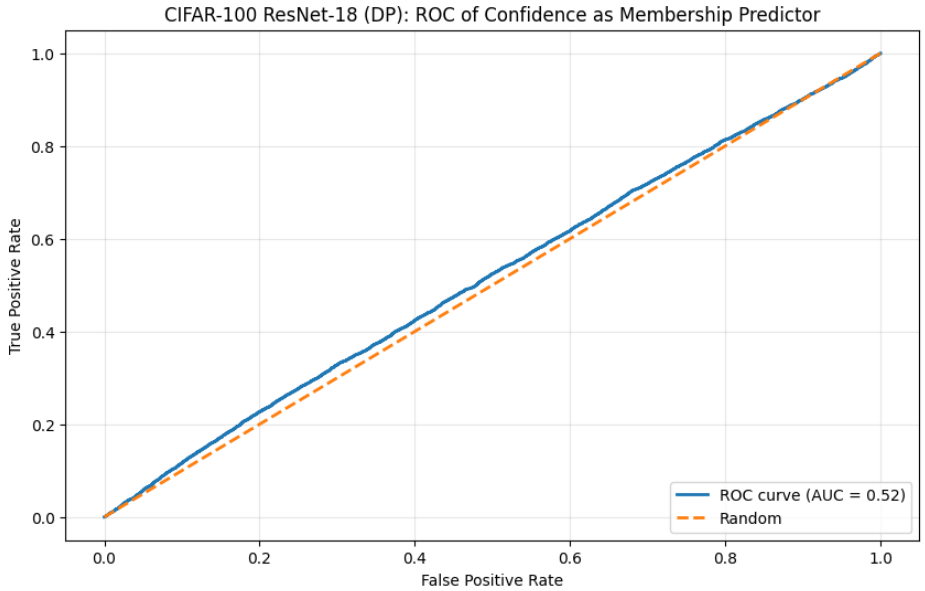


Figure A.74: ROC curve for confidence-based MIA (CIFAR-100 /w Transfer-Regularized model + DP + Epsilon $\varepsilon = 1.5$).

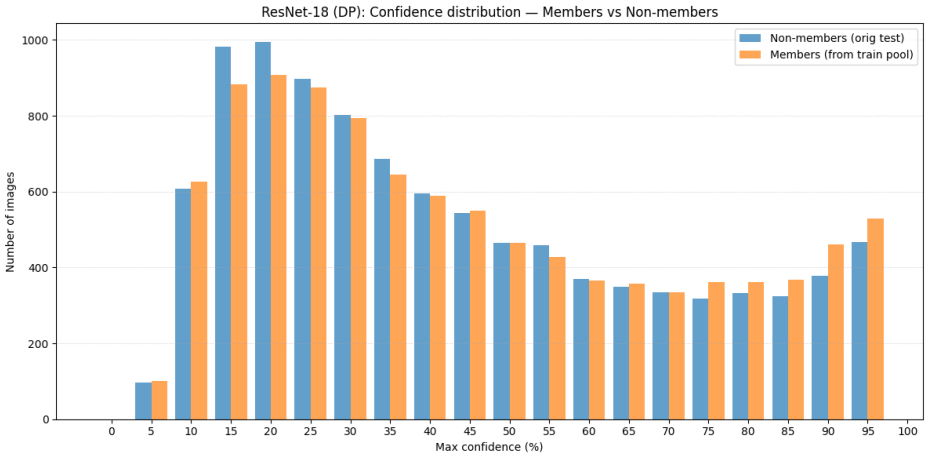


Figure A.75: Confidence distributions for members vs non-members (CIFAR-100 /w Transfer-Regularized model + DP + Epsilon $\varepsilon = 1.5$.

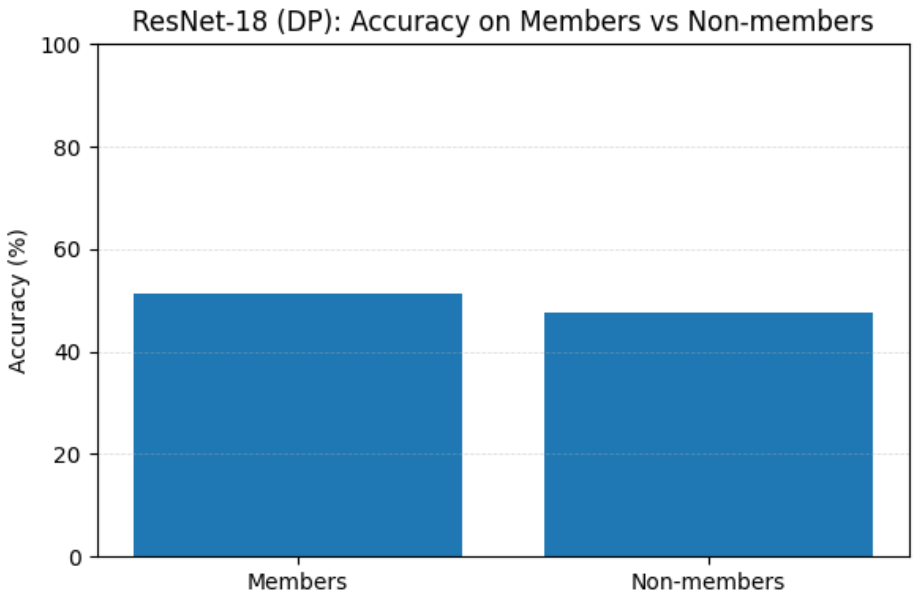
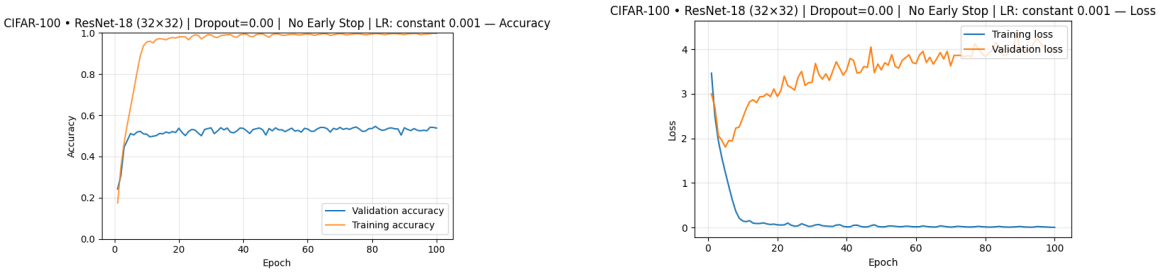


Figure A.76: Accuracy comparison for members and non-members (CIFAR-100 /w Transfer-Regularized model + DP + Epsilon $\varepsilon = 1.5$).

A.2.2 Resnet

Baseline

LINK: `https://colab.research.google.com/drive/1dRk9wGILTE5DCWq9G8Ic4ZhDnYf_kAOT?usp=sharing`



(a) Accuracy. **(b)** Loss.

Figure A.77: Training and validation performance for the CIFAR-100 ResNet-18 (non-pretrained) baseline model.

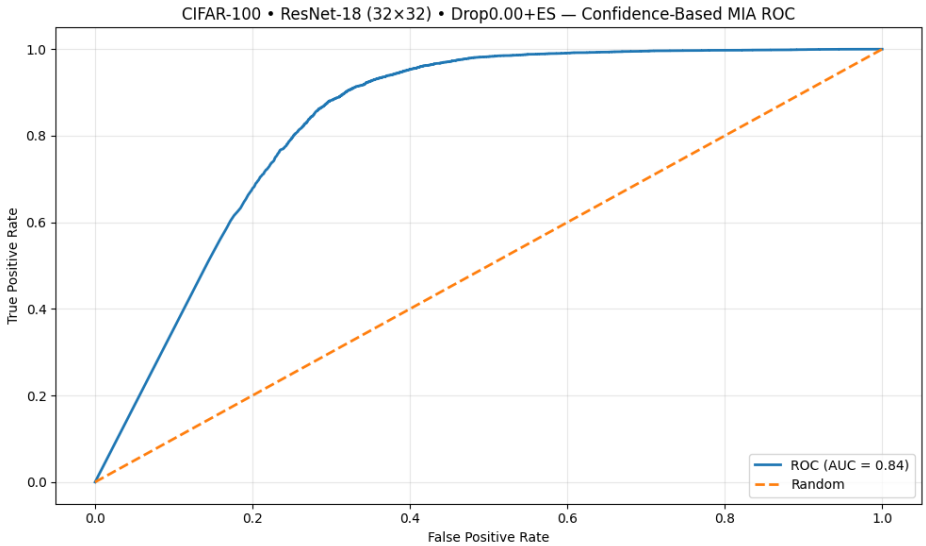


Figure A.78: ROC curve for confidence-based MIA (CIFAR-100 ResNet-18 non-pretrained baseline model).

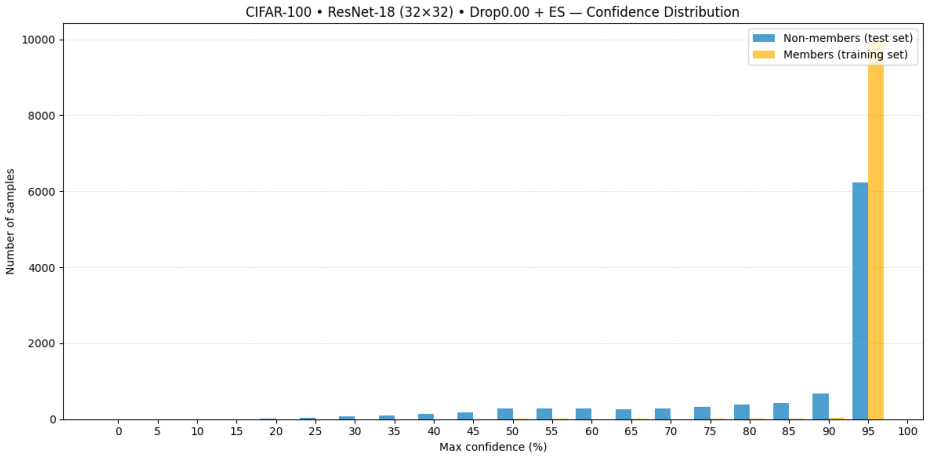


Figure A.79: Confidence distributions for members vs non-members (CIFAR-100 ResNet-18 non-pretrained baseline model).

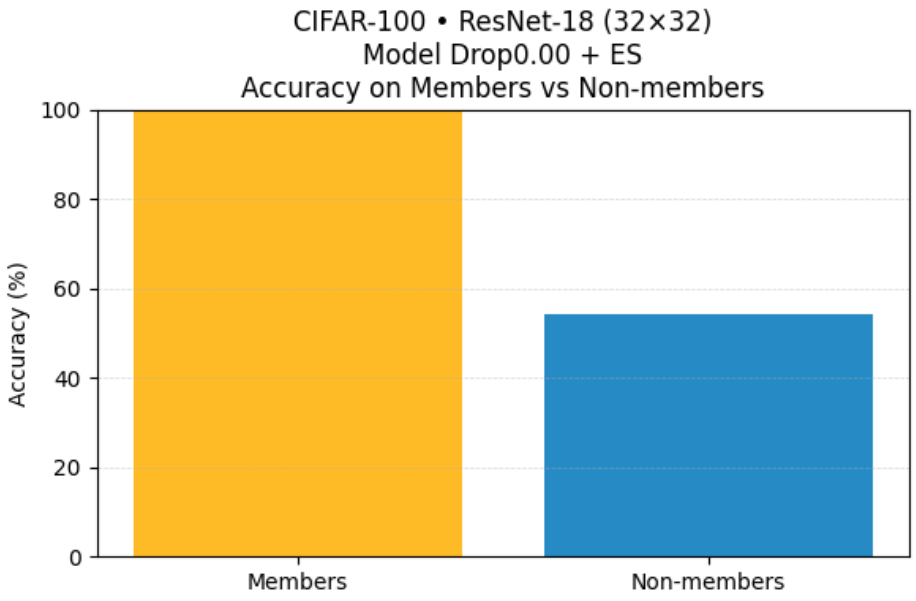
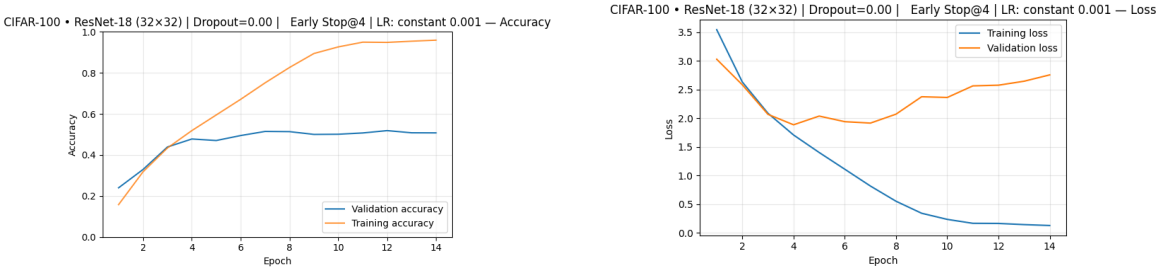


Figure A.80: Accuracy comparison for members and non-members (CIFAR-100 ResNet-18 non-pretrained baseline model).

Regularised



(a) Accuracy. **(b)** Loss.

Figure A.81: Training and validation performance for the CIFAR-100 ResNet-18 (non-pretrained) Regularised model.

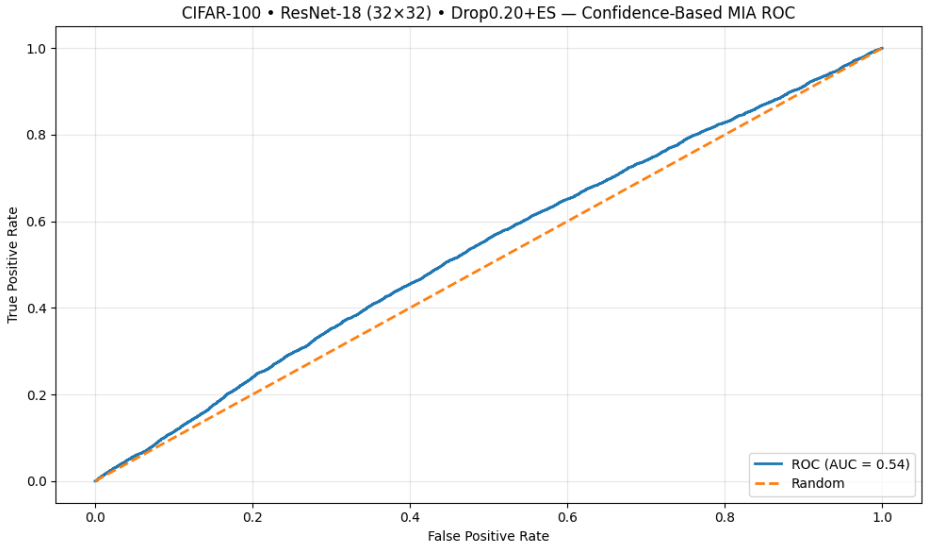


Figure A.82: ROC curve for confidence-based MIA (CIFAR-100 ResNet-18 non-pretrained Regularised model).

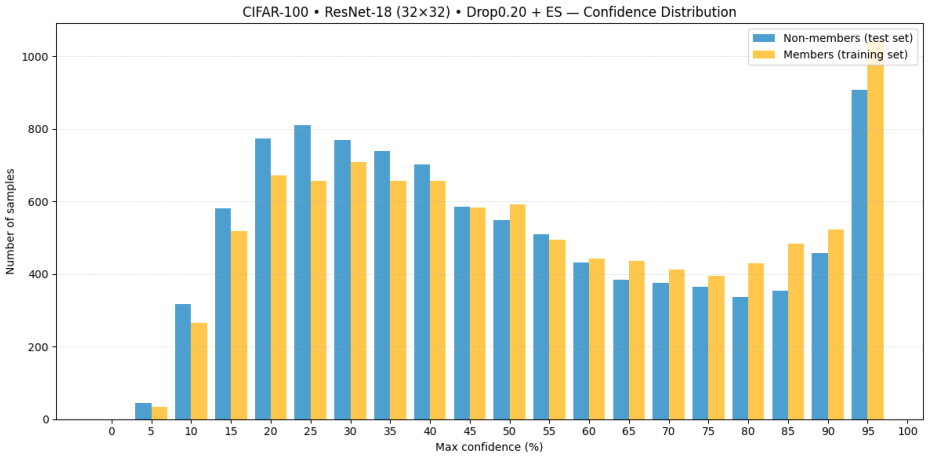


Figure A.83: Confidence distributions for members vs non-members (CIFAR-100 ResNet-18 non-pretrained Regularised model).

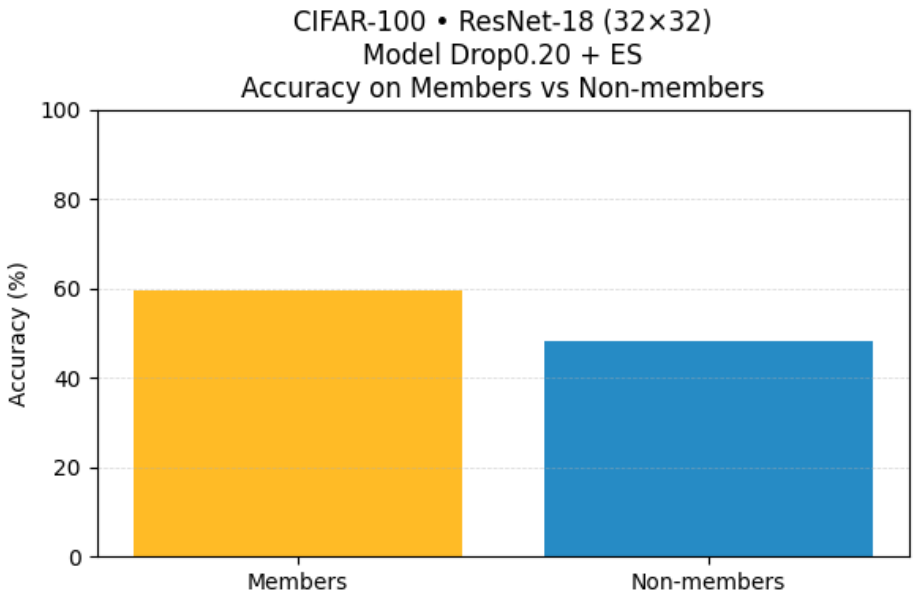
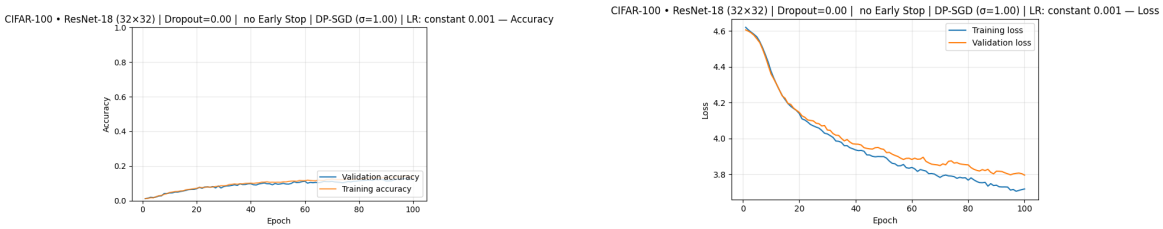


Figure A.84: Accuracy comparison for members and non-members (CIFAR-100 ResNet-18 non-pretrained Regularised model).

DP no Regularisation with Epsilon ε = 1.0

LINK: `https://colab.research.google.com/drive/1-5Pl7DtmiuFBrT6ZsdajEK4YoyaLRjo4?usp=sharing`



(a) Accuracy. **(b)** Loss.

Figure A.85: Training and validation performance for the CIFAR-100 ResNet-18 (non-pretrained) with DP No Regularised model.

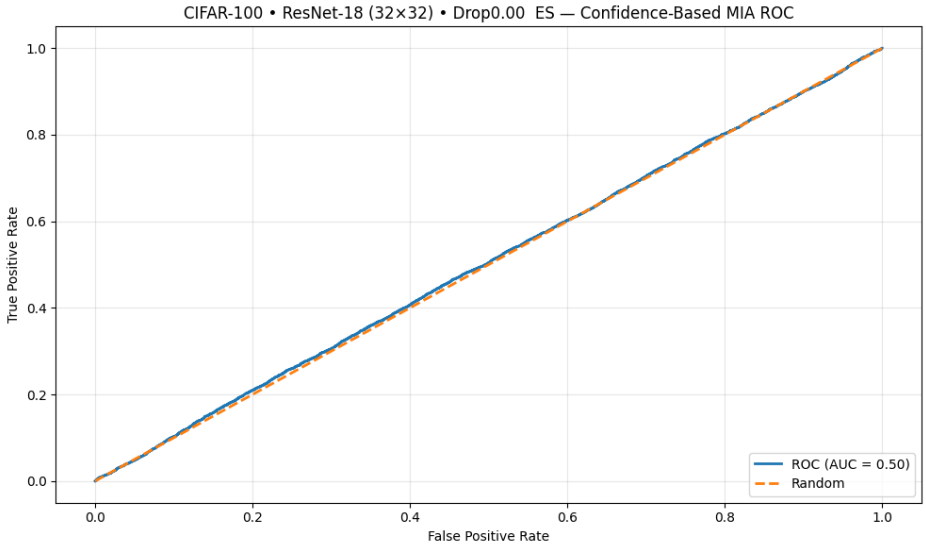


Figure A.86: ROC curve for confidence-based MIA (CIFAR-100 ResNet-18 non-pretrained with DP No Regularised model).

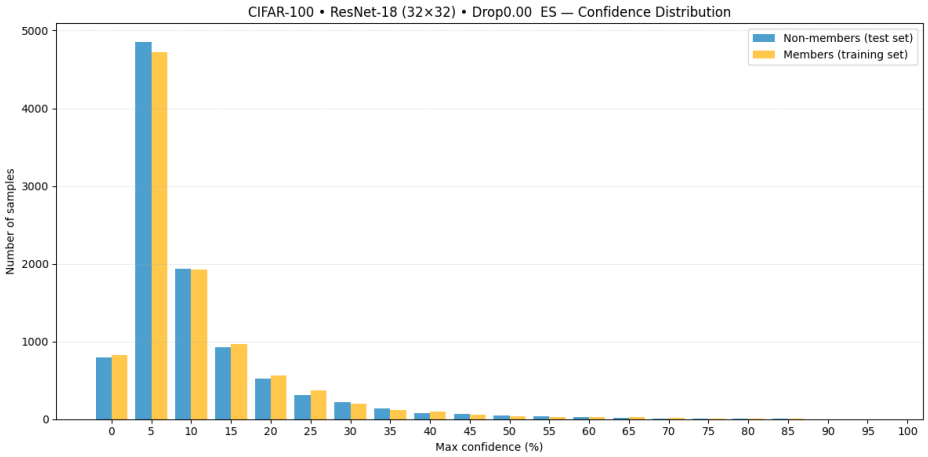


Figure A.87: Confidence distributions for members vs non-members (CIFAR-100 ResNet-18 non-pretrained with DP No Regularised model).

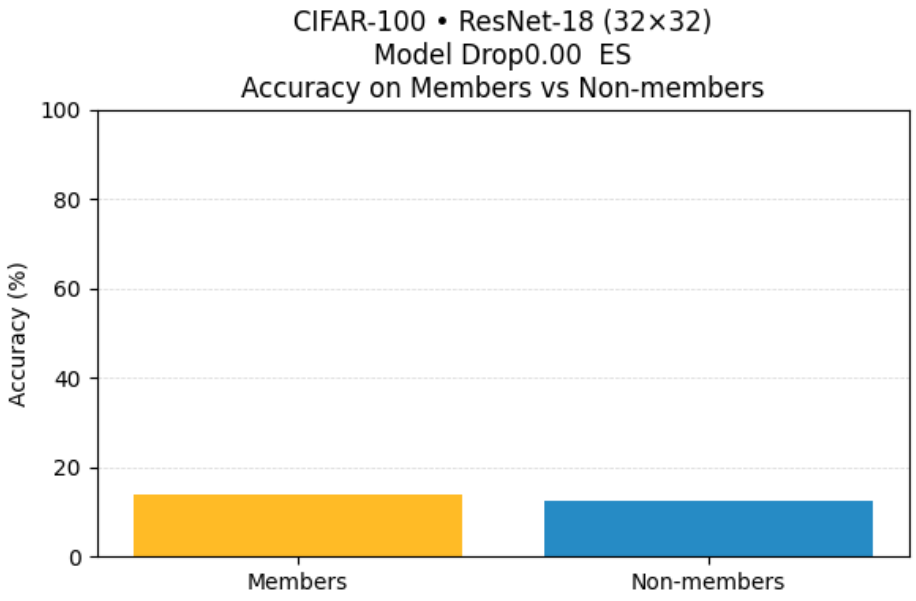
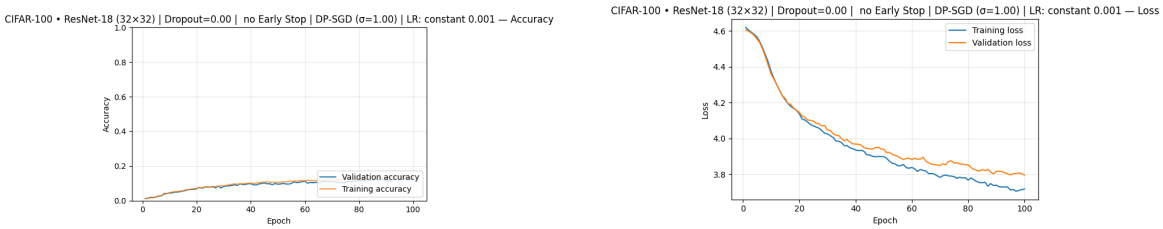


Figure A.88: Accuracy comparison for members and non-members (CIFAR-100 ResNet-18 non-pretrained with DP No Regularised model).

DP + Regularisation with Epsilon ε = 1.0

LINK: `https://colab.research.google.com/drive/1dRk9wGILTE5DCWq9G8Ic4ZhDnYf_kAOT?usp=sharing`



(a) Accuracy. **(b)** Loss.

Figure A.89: Training and validation performance for the CIFAR-100 ResNet-18 (non-pretrained) with DP Regularised model.

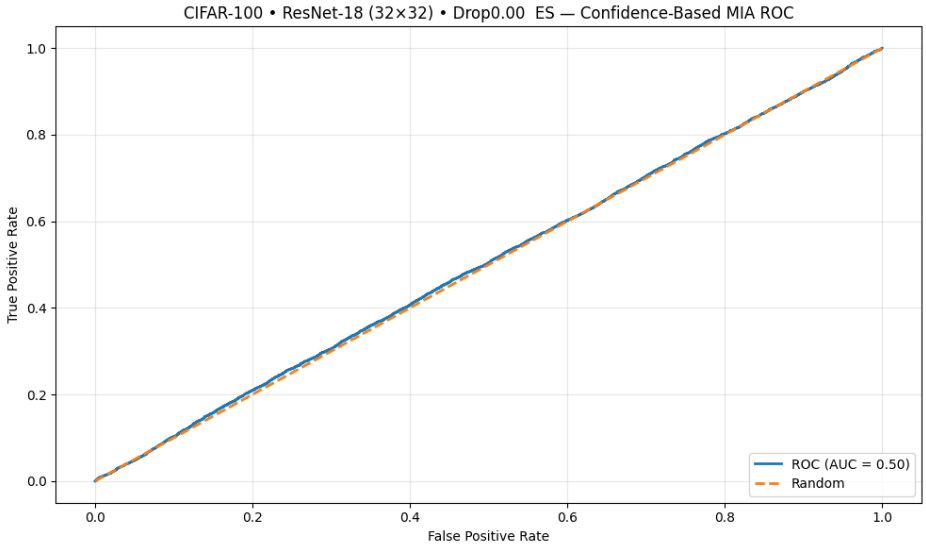


Figure A.90: ROC curve for confidence-based MIA (CIFAR-100 ResNet-18 non-pretrained with DP Regularised model).

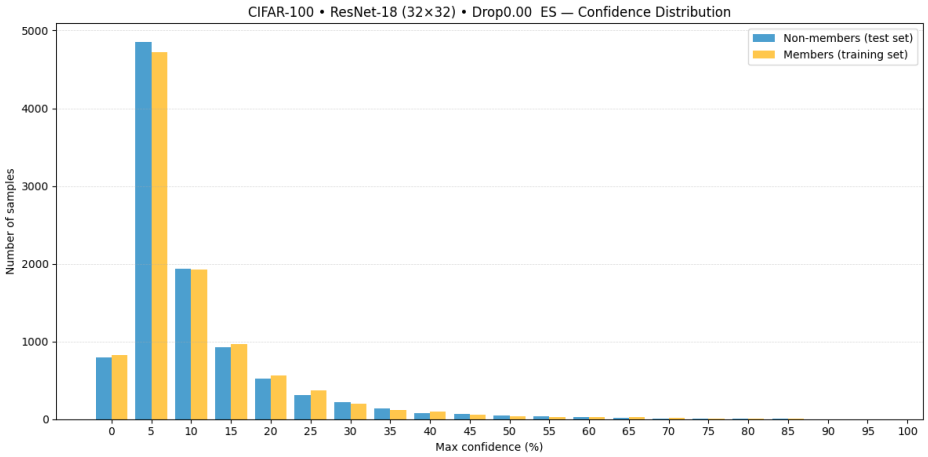


Figure A.91: Confidence distributions for members vs non-members (CIFAR-100 ResNet-18 non-pretrained with DP Regularised model).

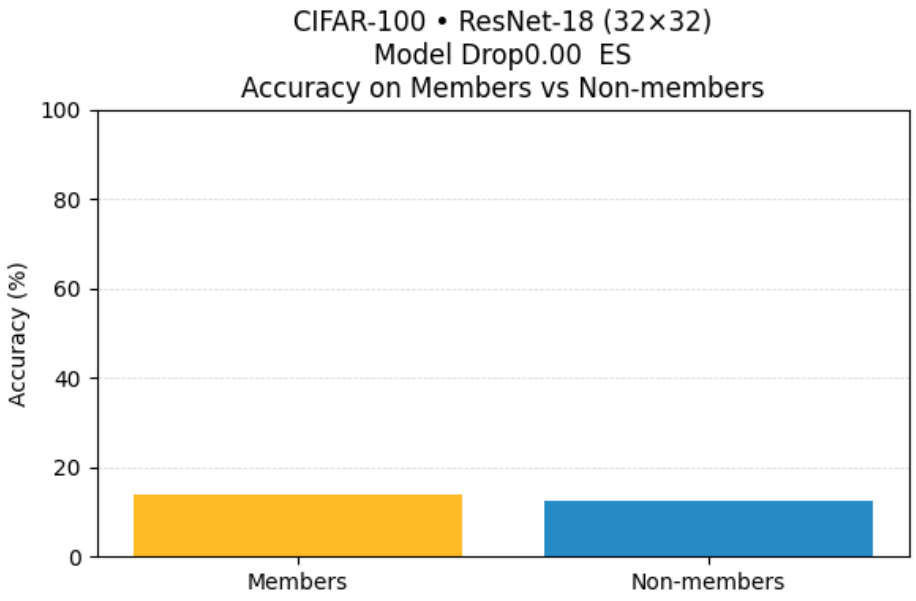
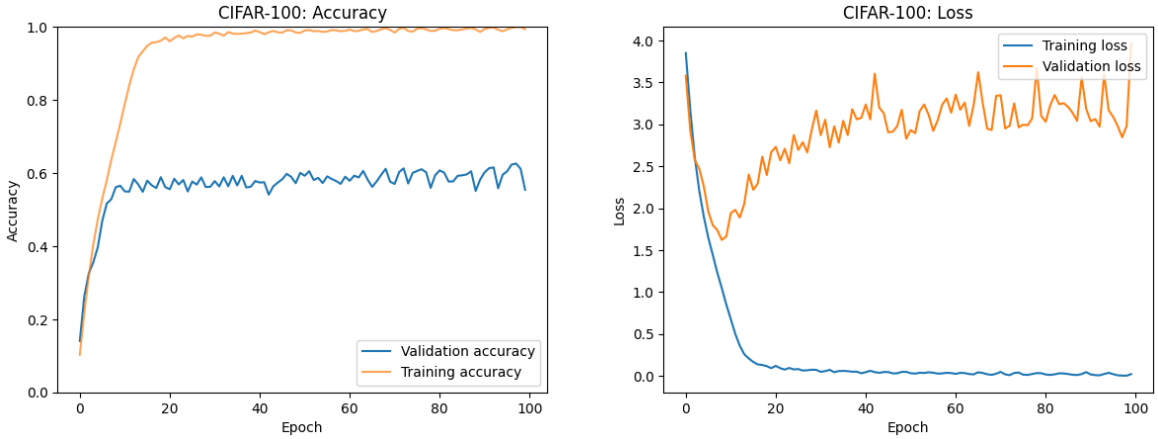


Figure A.92: Accuracy comparison for members and non-members (CIFAR-100 ResNet-18 non-pretrained with DP Regularised model).

A.2.3 WideResNet

Baseline

Link: `https://colab.research.google.com/drive/1e6jZ2D6U2yWuYXV57_gJbWXCBIcg3I8z?usp=sharing`



(a) Accuracy. **(b)** Loss.

Figure A.93: Training and validation performance for the CIFAR-100 WideResNet Baseline model.

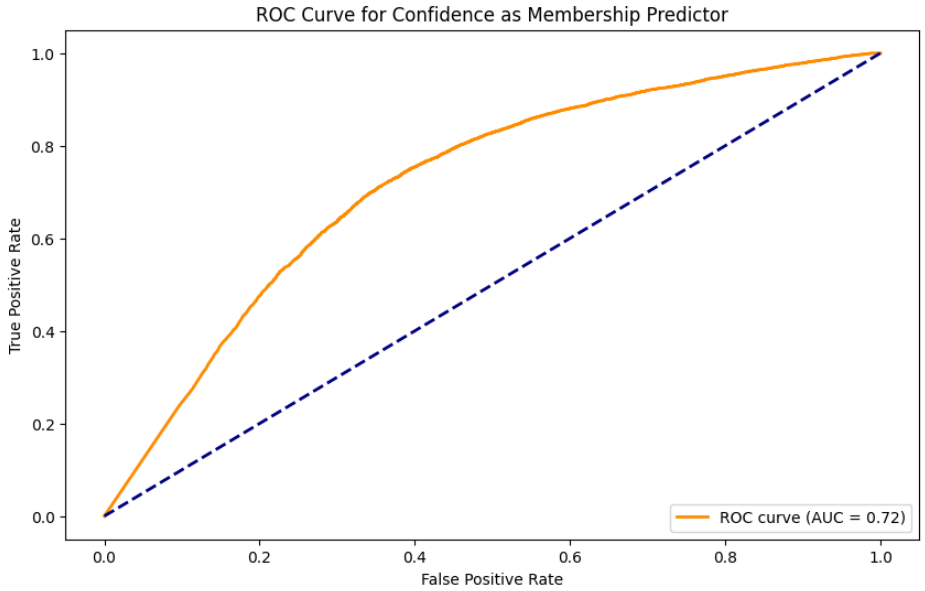


Figure A.94: ROC curve for confidence-based MIA (CIFAR-100 WideResNet Baseline model).

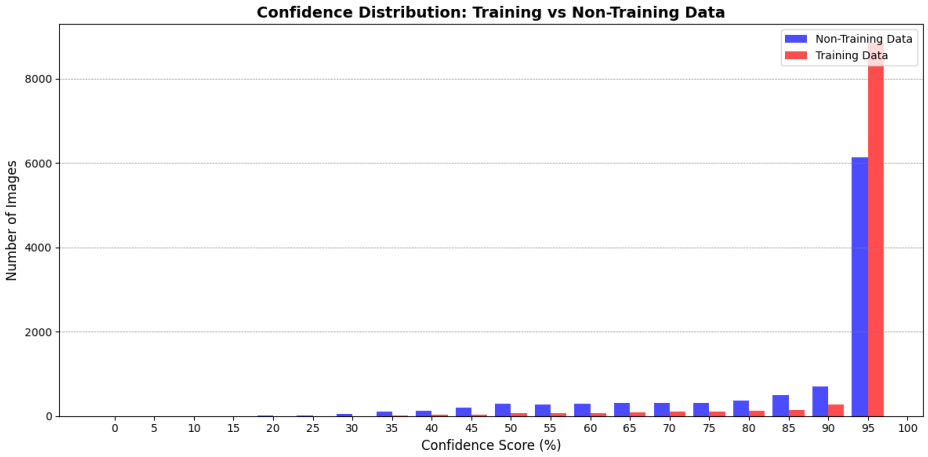


Figure A.95: Confidence distributions for members vs non-members (CIFAR-100 WideResNet Baseline model.

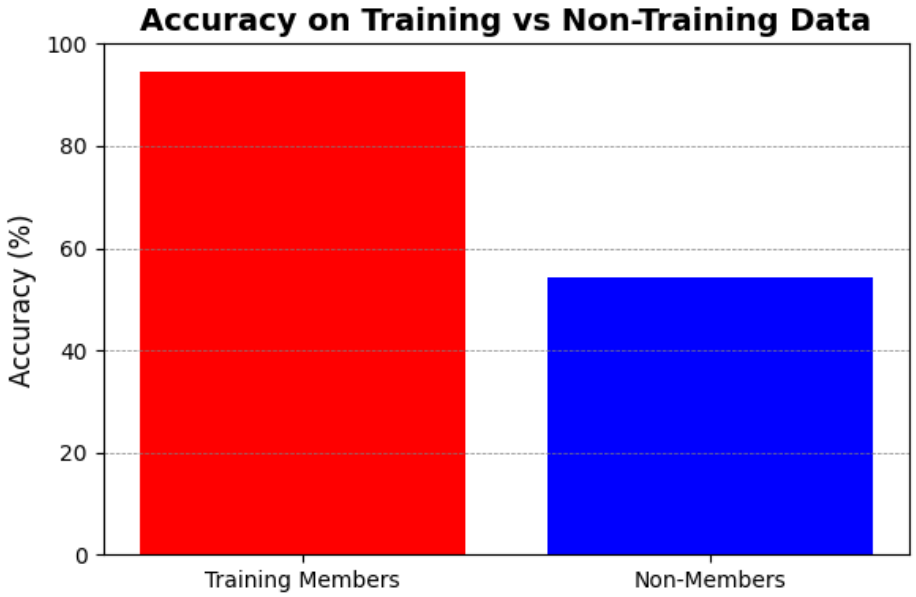
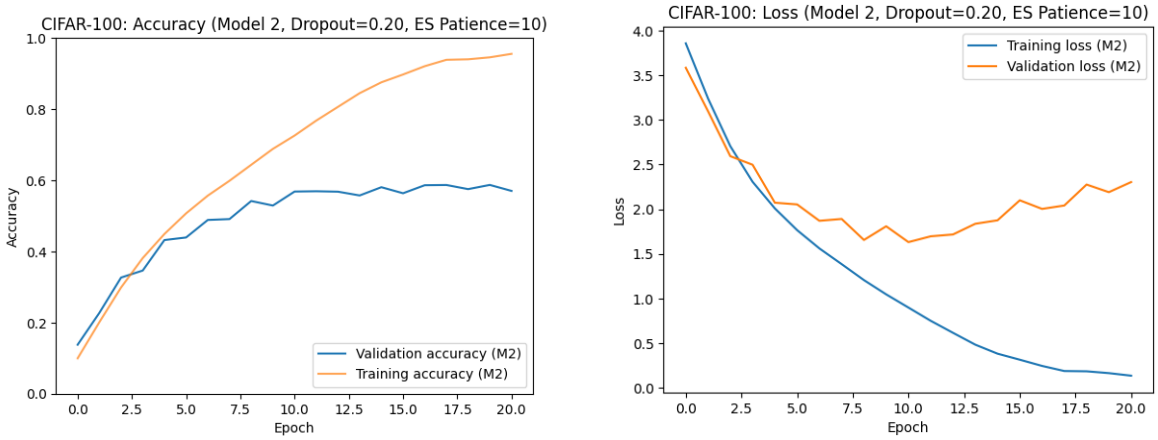


Figure A.96: Accuracy comparison for members and non-members (CIFAR-100 WideResNet Baseline model).

Regularised (dropout + early stopping)

Link: `https://colab.research.google.com/drive/1pMOX9xpGaaBhH3aSk-Y-_I-lnfHPjUP7?usp=sharing`



(a) Accuracy.

(b) Loss.

Figure A.97: Training and validation performance for the CIFAR-100 WideResNet Regularised model.

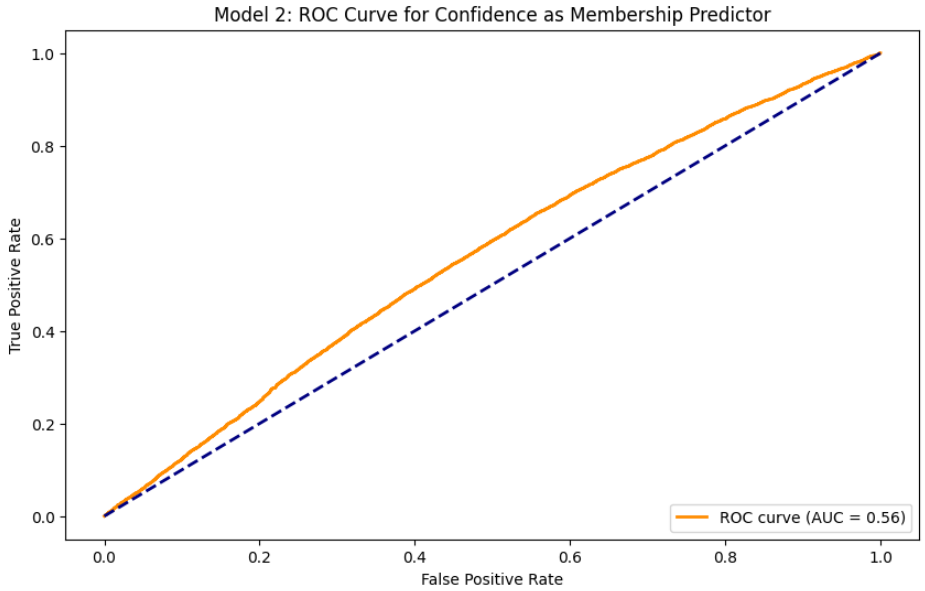


Figure A.98: ROC curve for confidence-based MIA (CIFAR-100 WideResNet Regularised model).

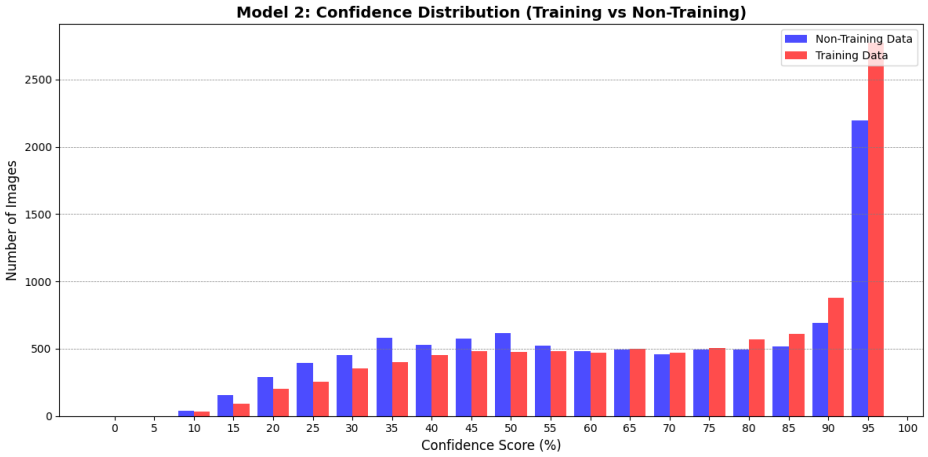


Figure A.99: Confidence distributions for members vs non-members (CIFAR-100 WideResNet Regularised model.

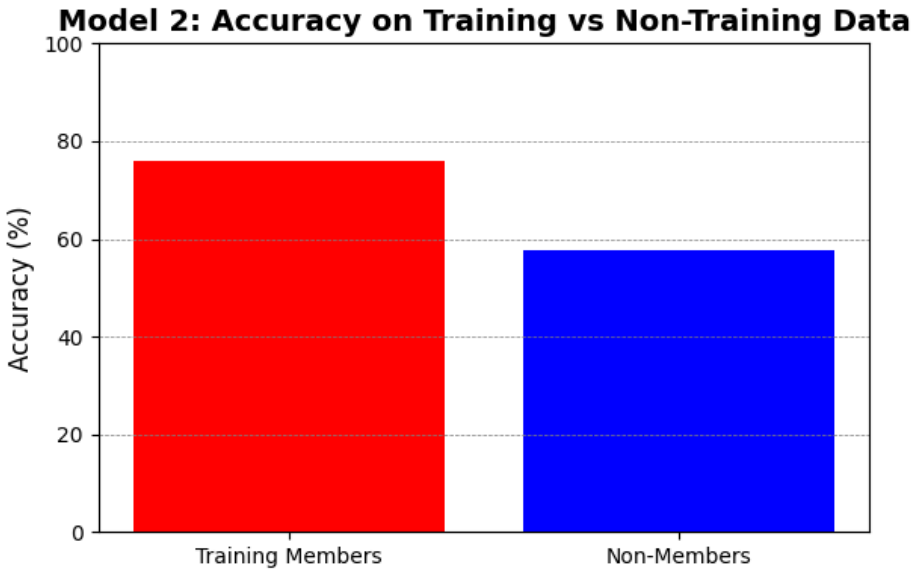
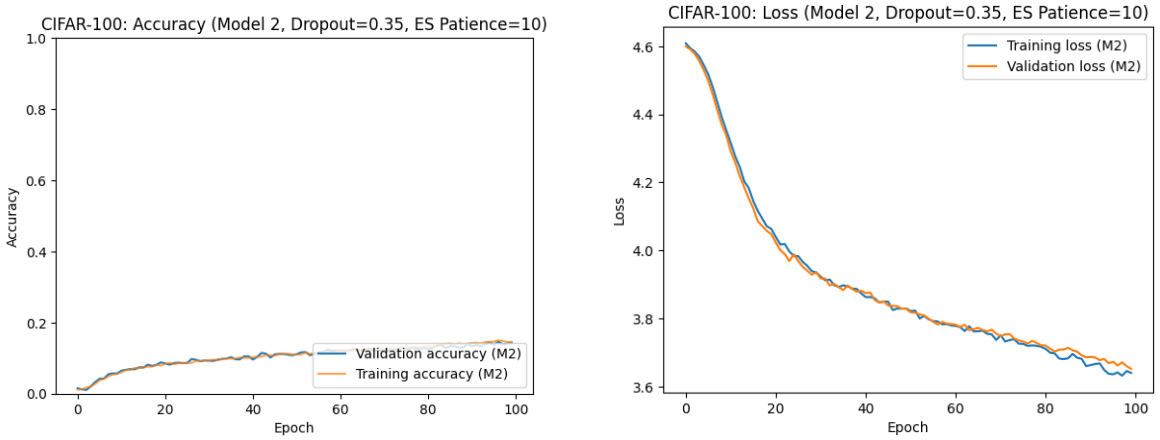


Figure A.100: Accuracy comparison for members and non-members (CIFAR-100 WideResNet Regularised model).

DP no Regularization with ε = 1.0)

Link: `https://colab.research.google.com/drive/10auKT1mywdK4RGxxrstq4mW6k5loq2Nw?usp=sharing`



(a) Accuracy.

(b) Loss.

Figure A.101: Training and validation performance for the CIFAR-100 DP-WideResNet-No-Regularised model.

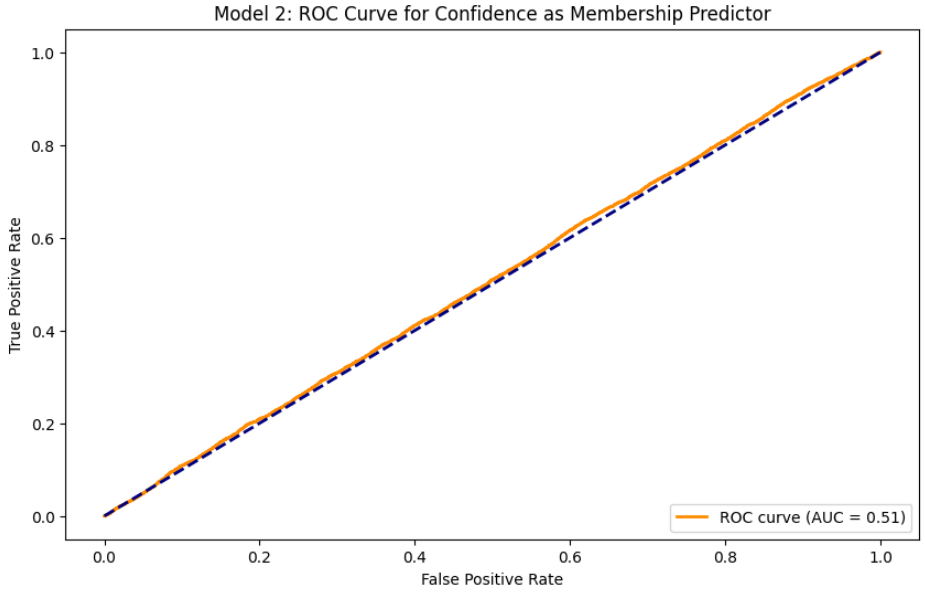


Figure A.102: ROC curve for confidence-based MIA (CIFAR-100 DP-WideResNet-No-Regularised model).

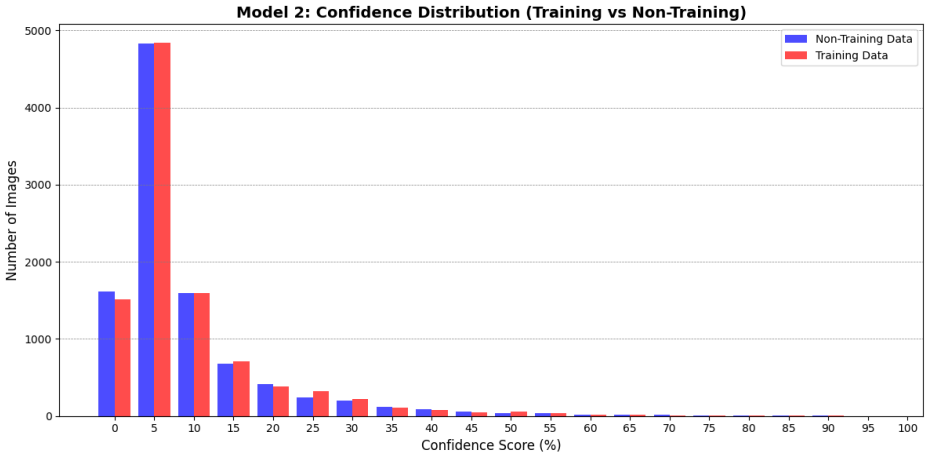


Figure A.103: Confidence distributions for members vs non-members (CIFAR-100 DP-WideResNet-No-Regularised model.

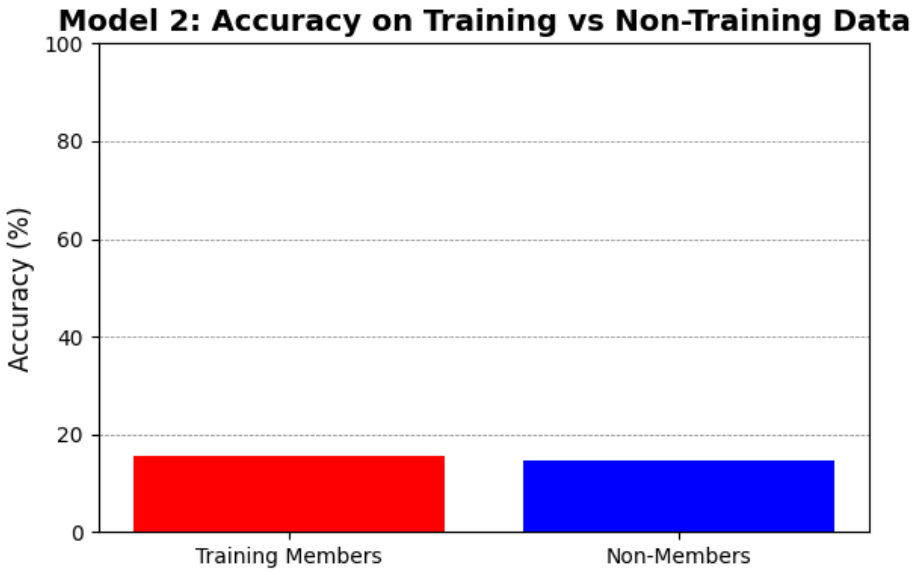
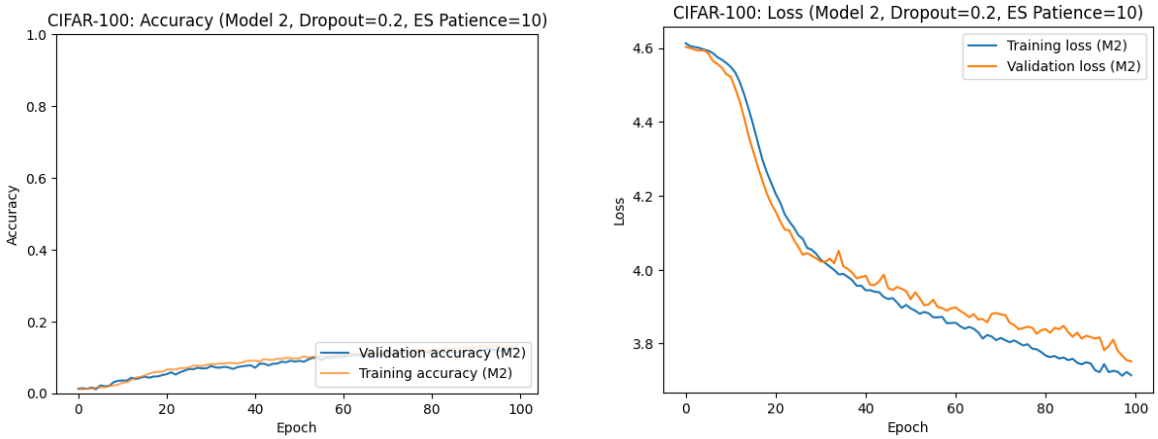


Figure A.104: Accuracy comparison for members and non-members (CIFAR-100 DP-WideResNet-No-Regularised model).

DP with ε = 1.0 + Regularization

Link: `https://colab.research.google.com/drive/11xeuAssXF6yDHlan8Xm4EnAbizE9q-Ad?usp=sharing`



(a) Accuracy.

(b) Loss.

Figure A.105: Training and validation performance for the CIFAR-100 DP-WideResNet-Regularised model.

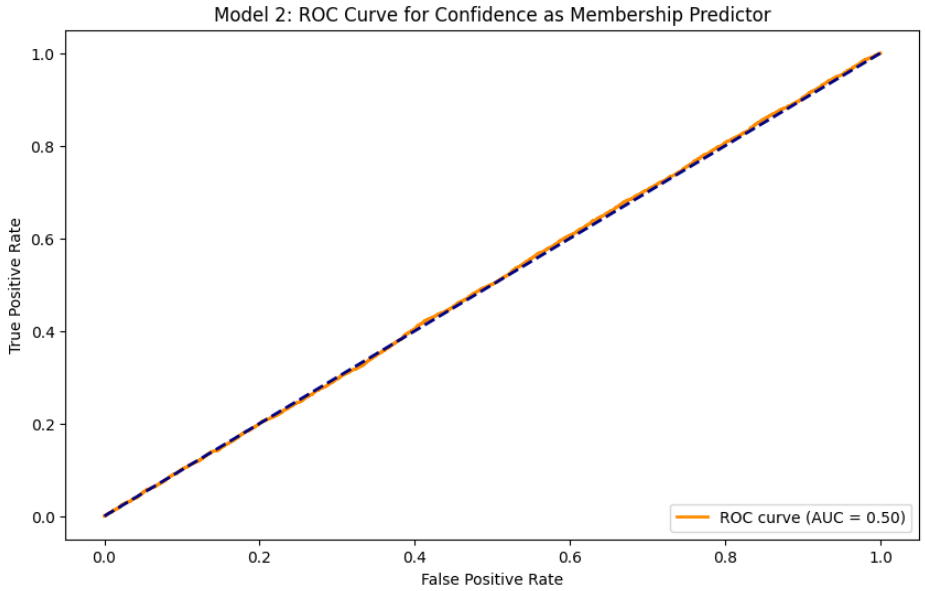


Figure A.106: ROC curve for confidence-based MIA (CIFAR-100 DP-WideResNet-Regularised model).

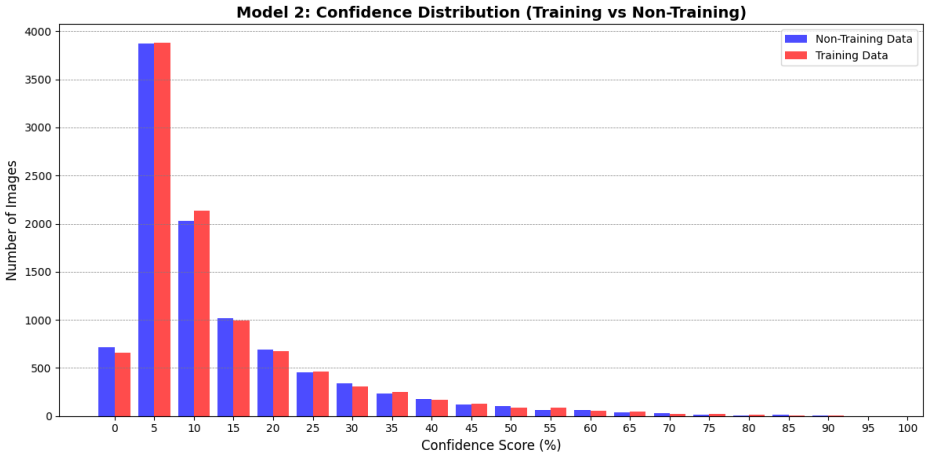


Figure A.107: Confidence distributions for members vs non-members (CIFAR-100 DP-WideResNet-Regularised model.

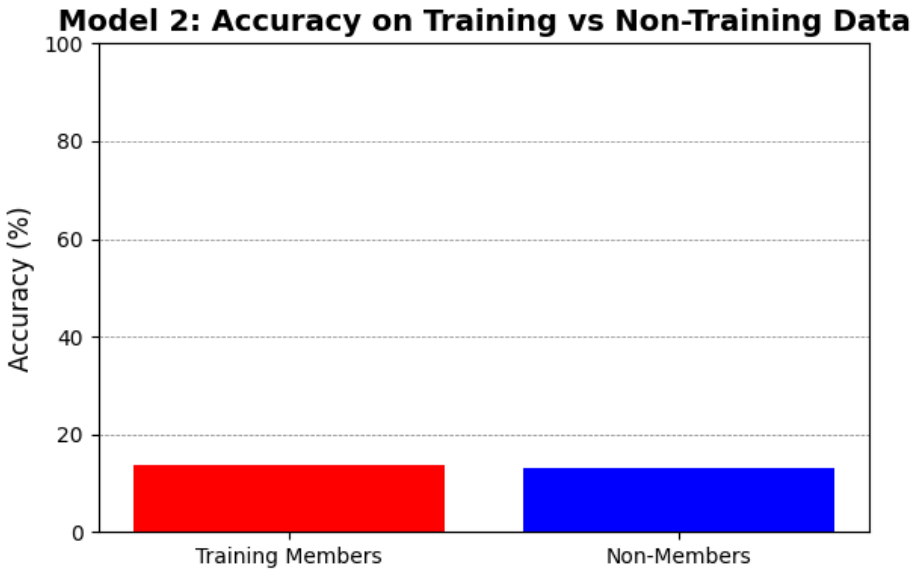
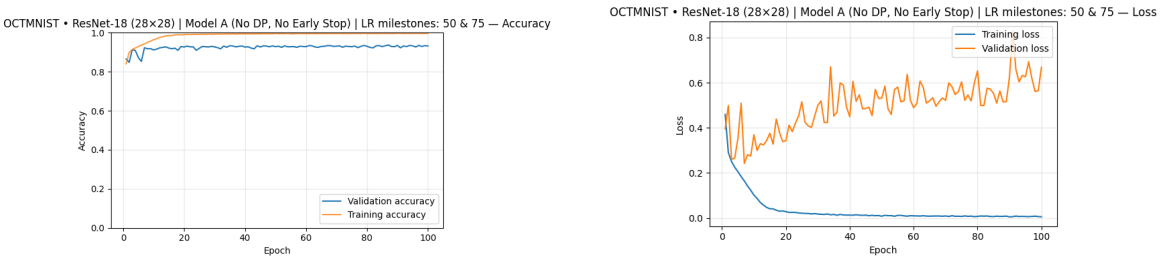


Figure A.108: Accuracy comparison for members and non-members (CIFAR-100 DP-WideResNet-Regularised model).

A.3 OCTMNIST

A.3.1 Baseline

LINK: `https://colab.research.google.com/drive/1-gHy8pKhAg10hY9RCA1ZD-6uoXqSSTwZ?usp=sharing`



(a) Accuracy. **(b)** Loss.

Figure A.109: Training and validation performance for the OCTMNIST baseline model.

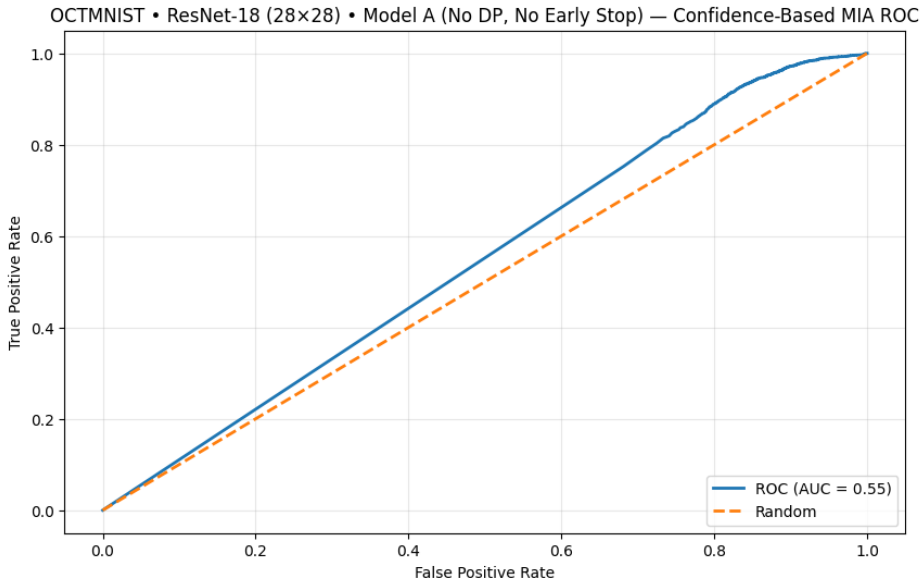


Figure A.110: ROC curve for confidence-based MIA (OCTMNIST baseline).

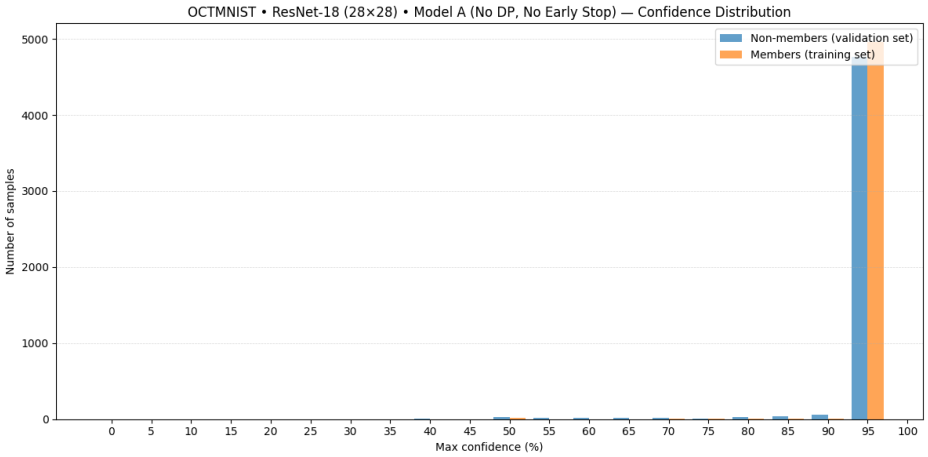


Figure A.111: Confidence distributions for members vs non-members (OCTMNIST baseline).

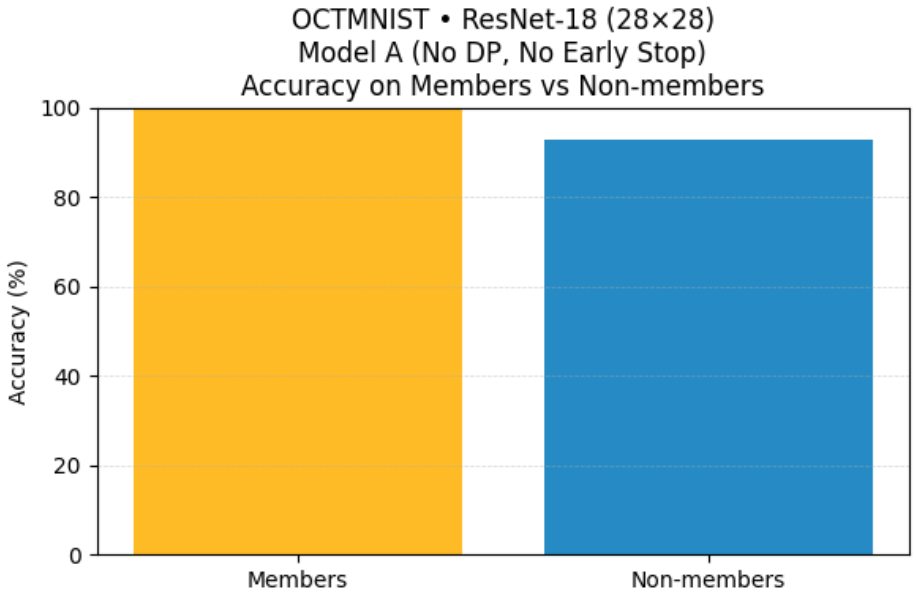
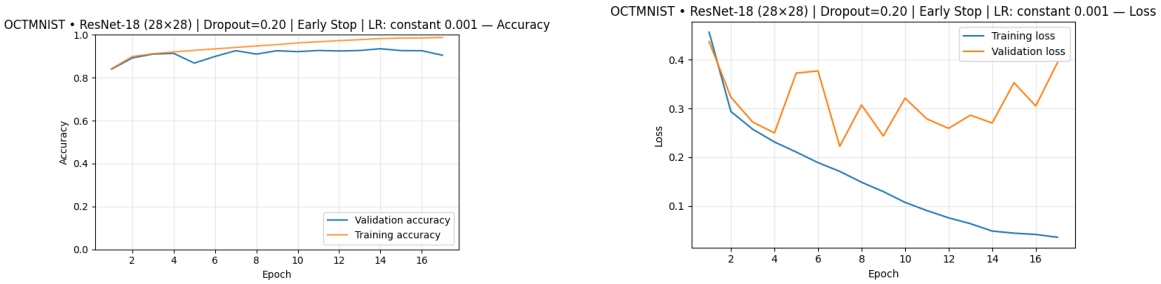


Figure A.112: Accuracy comparison for members and non-members (OCTMNIST baseline).

A.3.2 Regularised

LINK: `https://colab.research.google.com/drive/1SjQyLi_njIXlOGrEjPJbMq5jAGEd1psF?usp=sharing`



(a) Accuracy. **(b)** Loss.

Figure A.113: Training and validation performance for the OCTMNIST Regularised model.

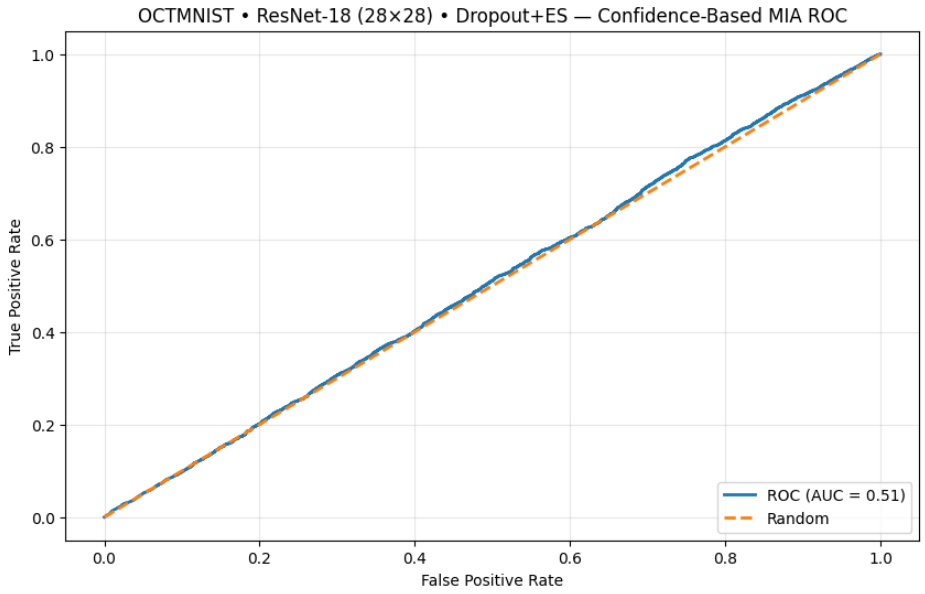


Figure A.114: ROC curve for confidence-based MIA (OCTMNIST Regularised).

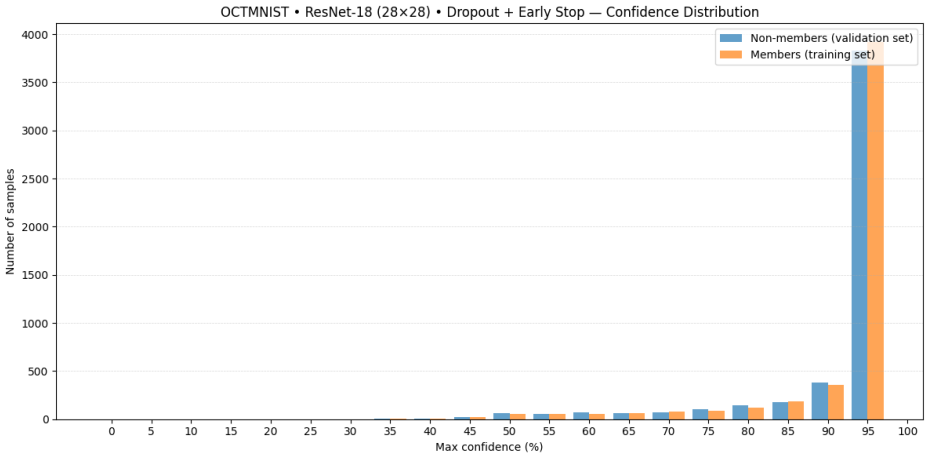


Figure A.115: Confidence distributions for members vs non-members (OCTMNIST Regularised).

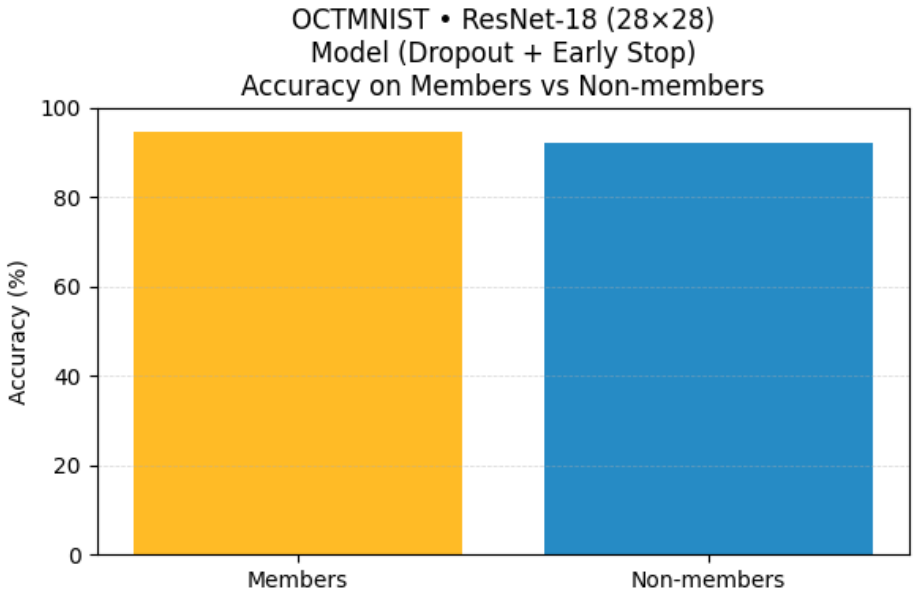
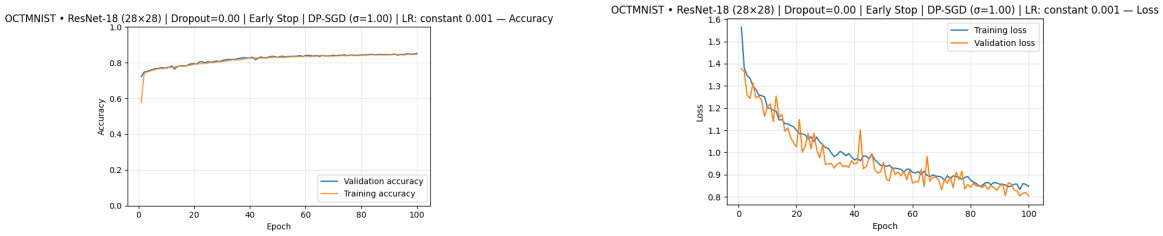


Figure A.116: Accuracy comparison for members and non-members (OCTMNIST Regularised).

A.3.3 DP no Regularization

LINK: `https://colab.research.google.com/drive/1aYACuvIKdXSGp5vgT2VFBgTiY_HxASnn?usp=sharing`

Epsilon ε = 1.0



(a) Accuracy. **(b)** Loss.

Figure A.117: Training and validation performance for the OCTMNIST No Regularisation with ε = 1.0 .

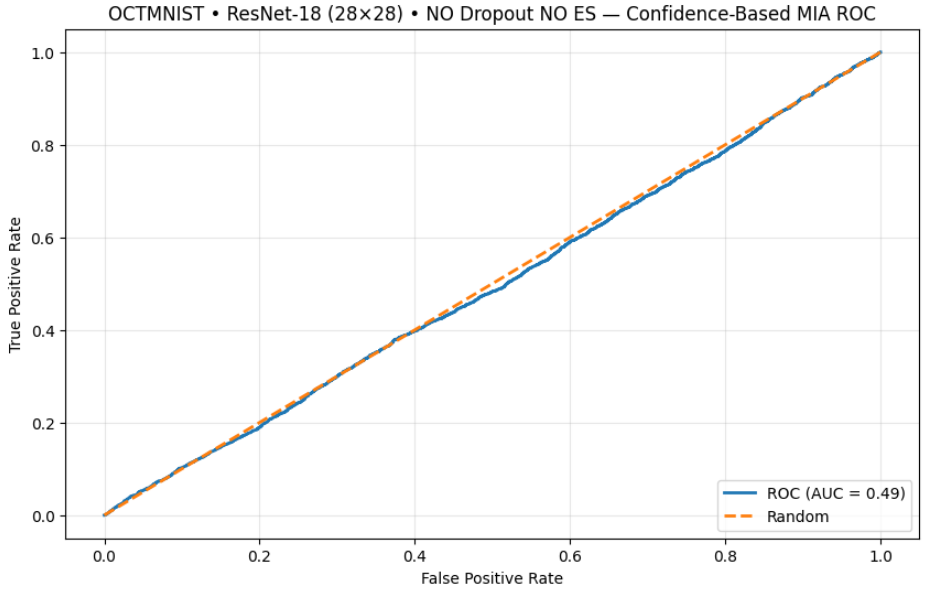


Figure A.118: ROC curve for confidence-based MIA (OCTMNIST No Regularisation with ε = 1.0).

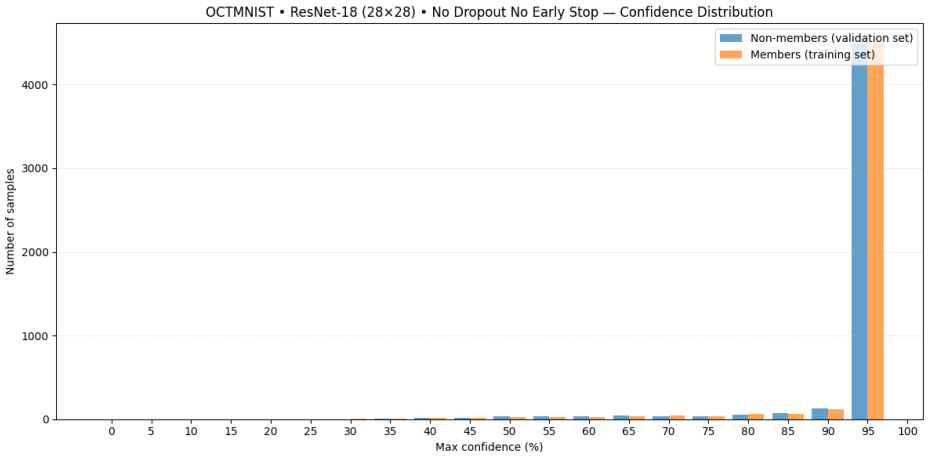


Figure A.119: Confidence distributions for members vs non-members (OCTMNIST No Regularisation with ε = 1.0).

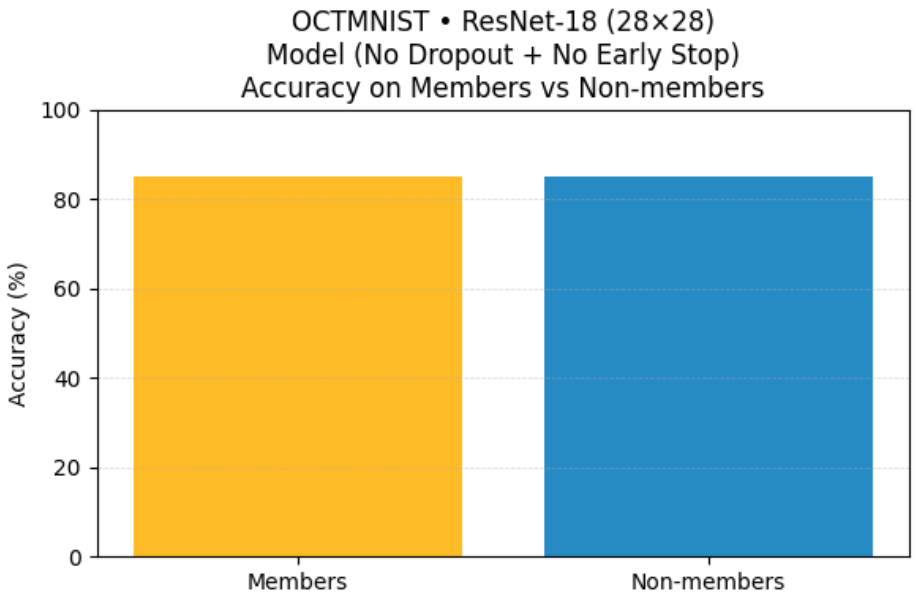
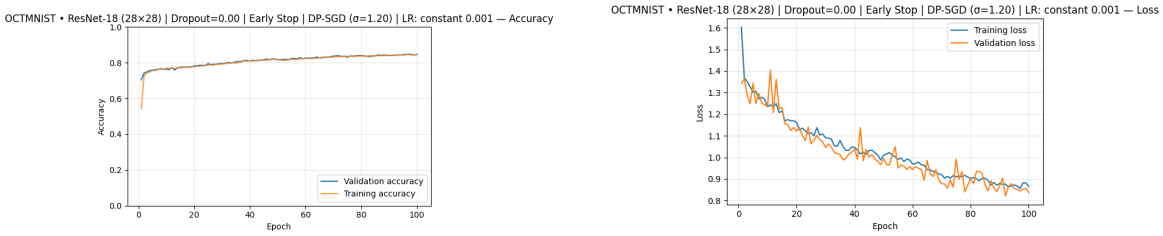


Figure A.120: Accuracy comparison for members and non-members (OCTMNIST No Regularisation with ε = 1.0).

Epsilon ε = 1.2



(a) Accuracy. **(b)** Loss.

Figure A.121: Training and validation performance for the OCTMNIST No Regularisation with ε = 1.2 .

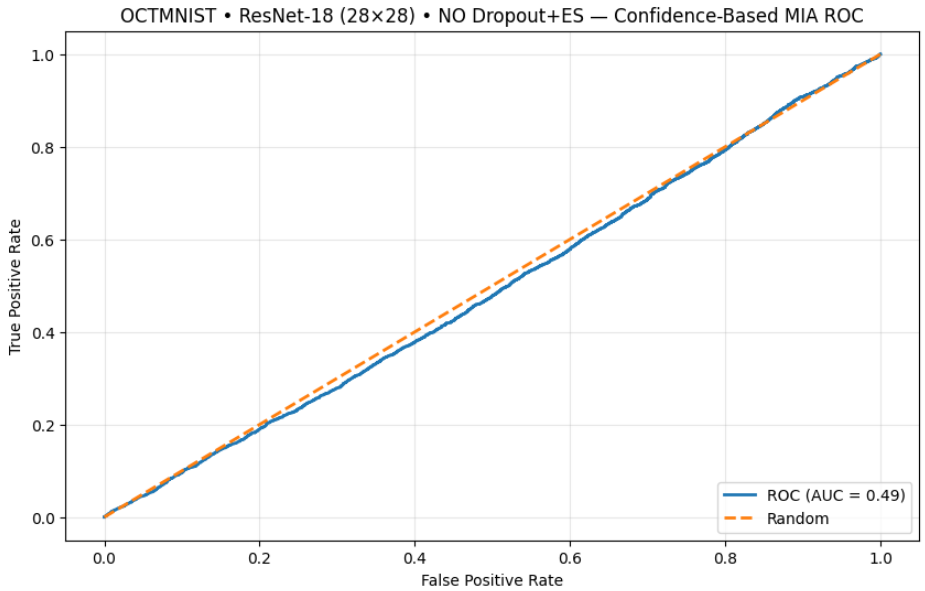


Figure A.122: ROC curve for confidence-based MIA (OCTMNIST No Regularisation with ε = 1.2).

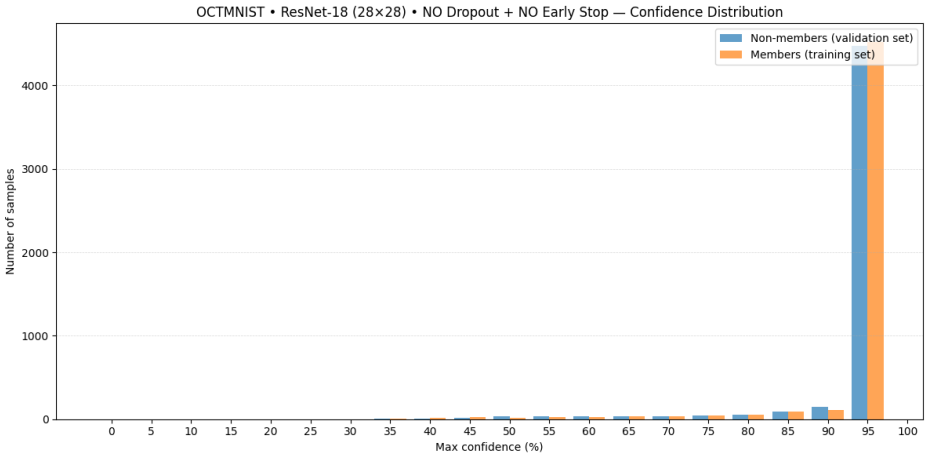


Figure A.123: Confidence distributions for members vs non-members (OCTMNIST No Regularisation with ε = 1.2).

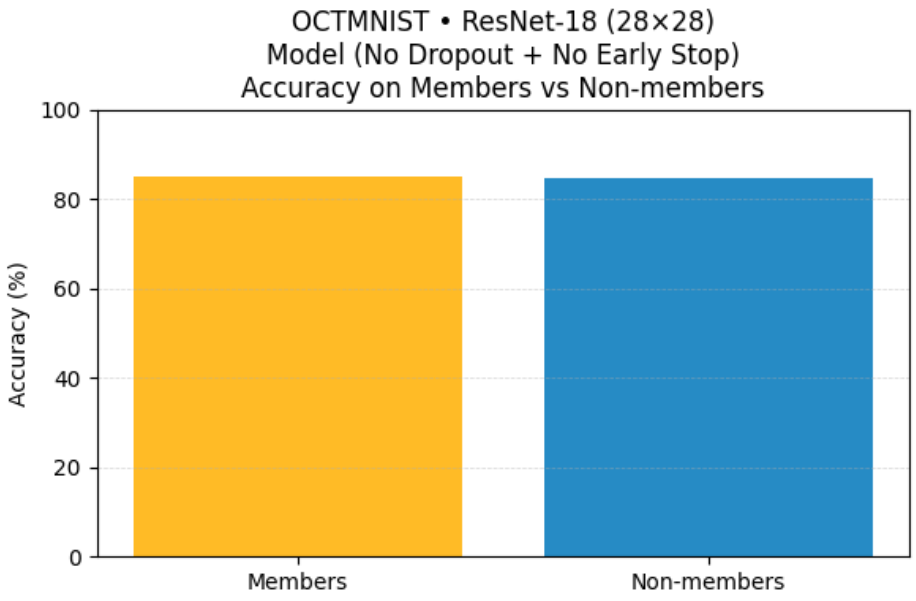
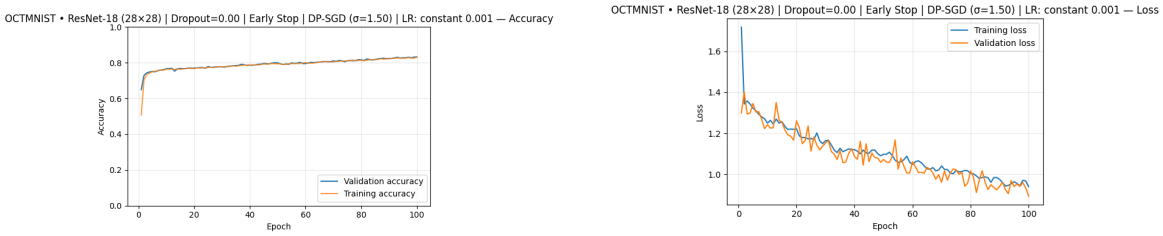


Figure A.124: Accuracy comparison for members and non-members (OCTMNIST No Regularisation with ε = 1.2).

Epsilon ε = 1.5



(a) Accuracy. **(b)** Loss.

Figure A.125: Training and validation performance for the OCTMNIST No Regularisation with ε = 1.5 .

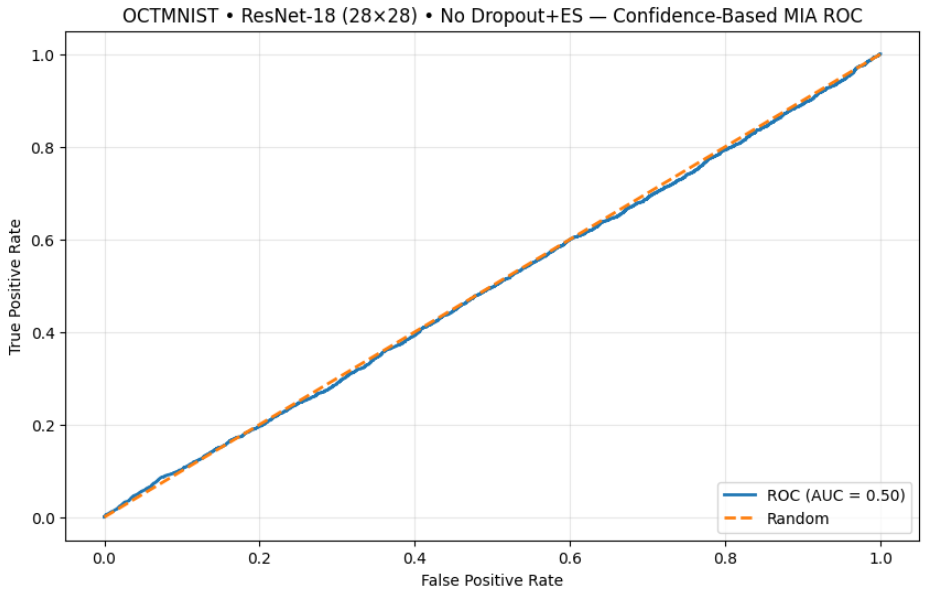


Figure A.126: ROC curve for confidence-based MIA (OCTMNIST No Regularisation with ε = 1.5).

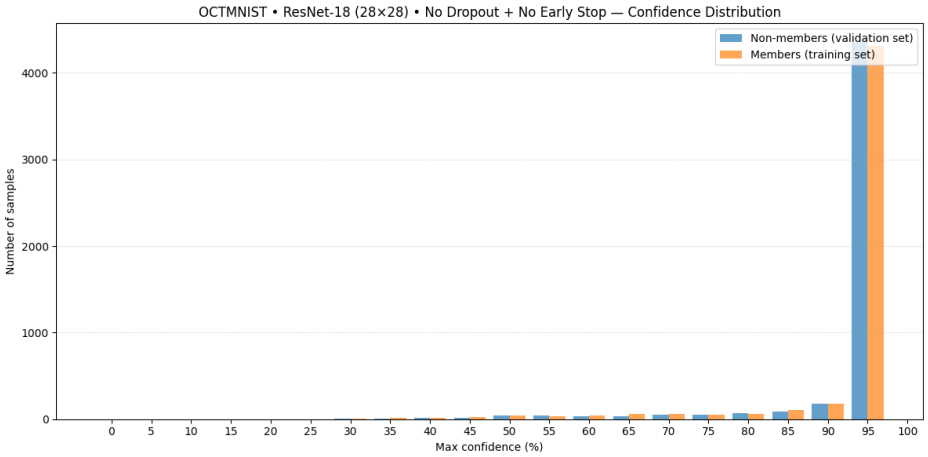


Figure A.127: Confidence distributions for members vs non-members (OCTMNIST No Regularisation with ε = 1.5).

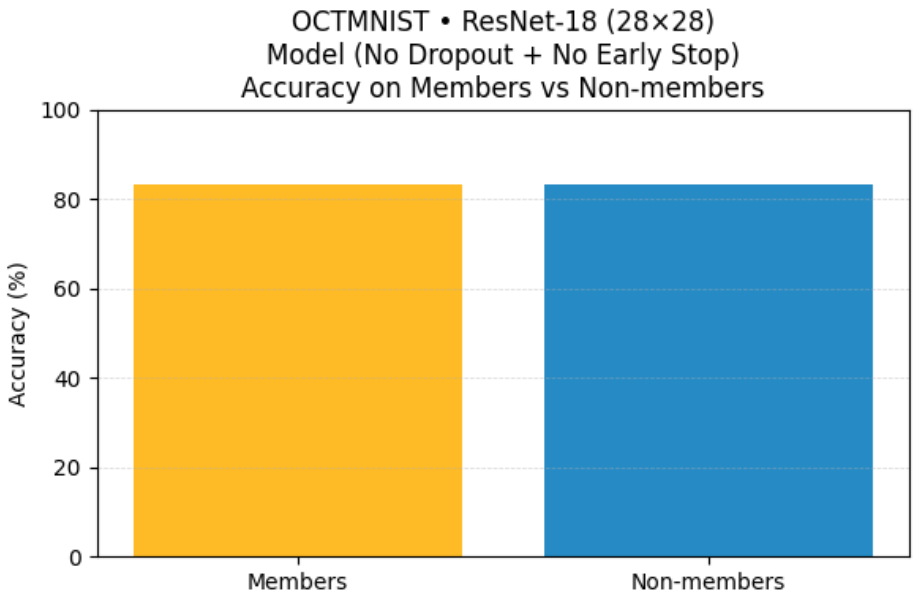
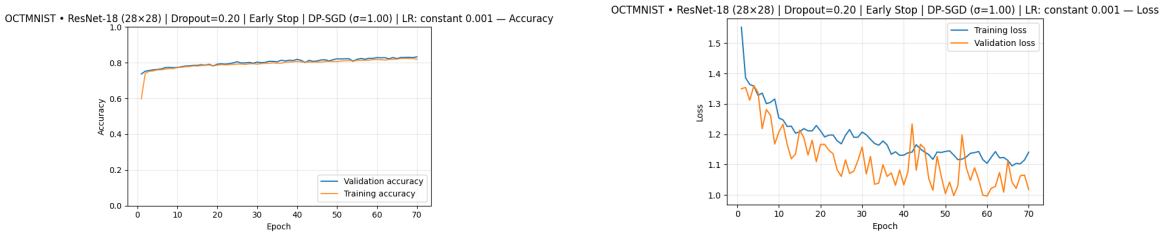


Figure A.128: Accuracy comparison for members and non-members (OCTMNIST No Regularisation with ε = 1.5).

A.3.4 DP + Regularisation

LINK:`https://colab.research.google.com/drive/1CeBQJ9Tm6JU5kEZVKBcFnCl7bQ5yjVyN?usp=sharing`

Epsilon ε = 1.0



(a) Accuracy. **(b)** Loss.

Figure A.129: Training and validation performance for the OCTMNIST + Regularisation with ε = 1.0 .

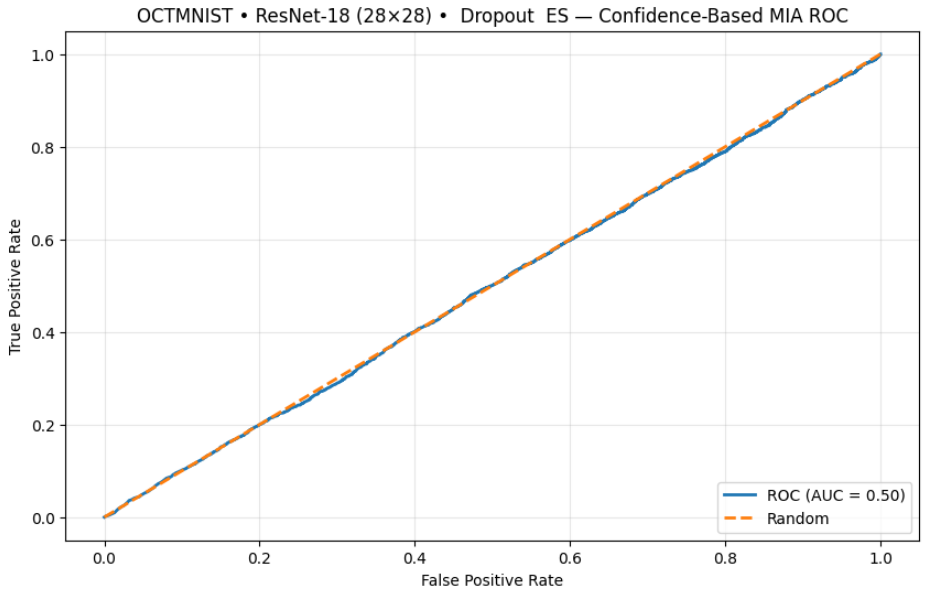


Figure A.130: ROC curve for confidence-based MIA (OCTMNIST + Regularisation with ε = 1.0).

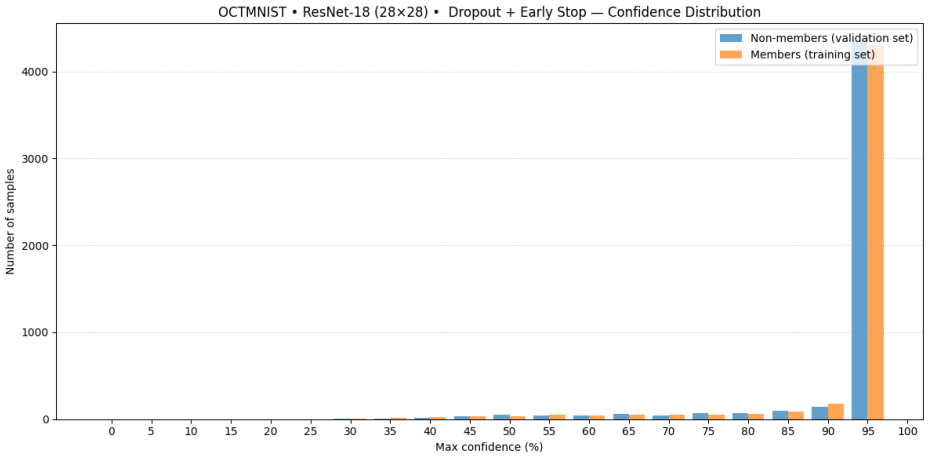


Figure A.131: Confidence distributions for members vs non-members (OCTMNIST + Regularisation with ε = 1.0).

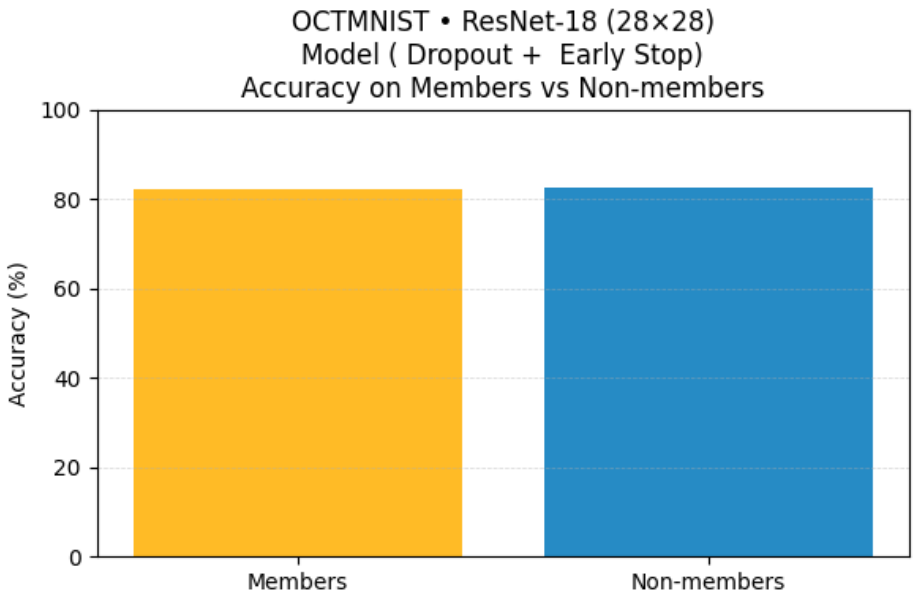
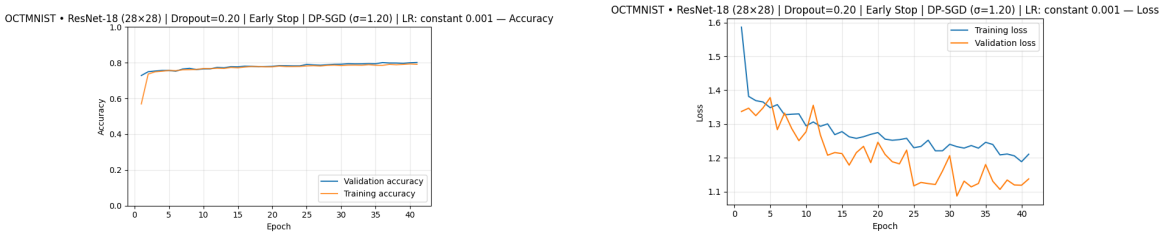


Figure A.132: Accuracy comparison for members and non-members (OCTMNIST + Regularisation with ε = 1.0).

Epsilon ε = 1.2



(a) Accuracy. **(b)** Loss.

Figure A.133: Training and validation performance for the OCTMNIST + Regularisation with ε = 1.2 .

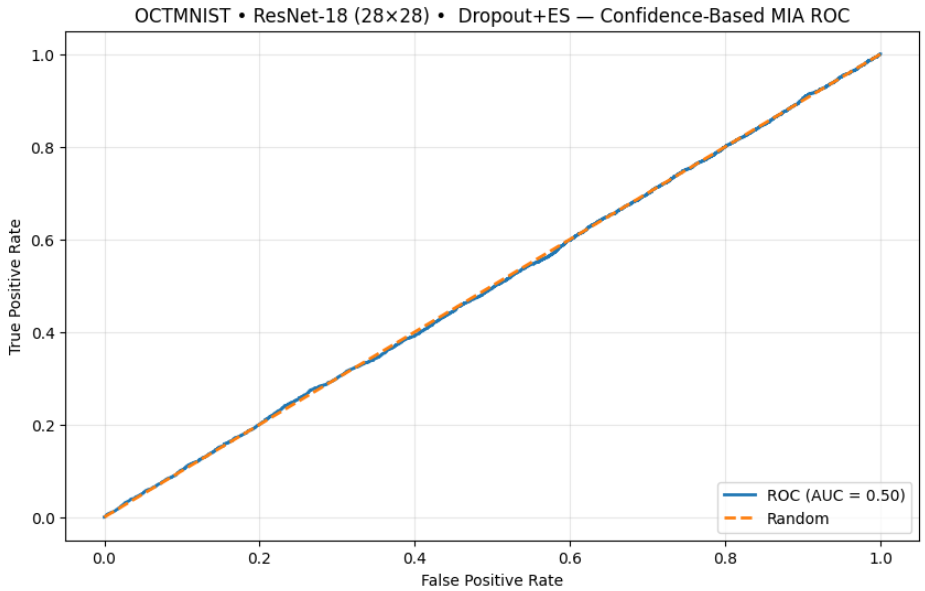


Figure A.134: ROC curve for confidence-based MIA (OCTMNIST + Regularisation with ε = 1.2).

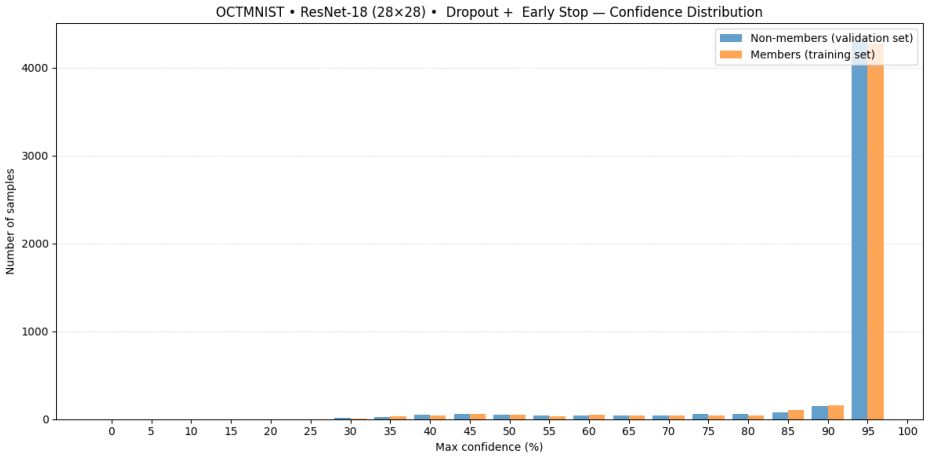


Figure A.135: Confidence distributions for members vs non-members (OCTMNIST + Regularisation with ε = 1.2).

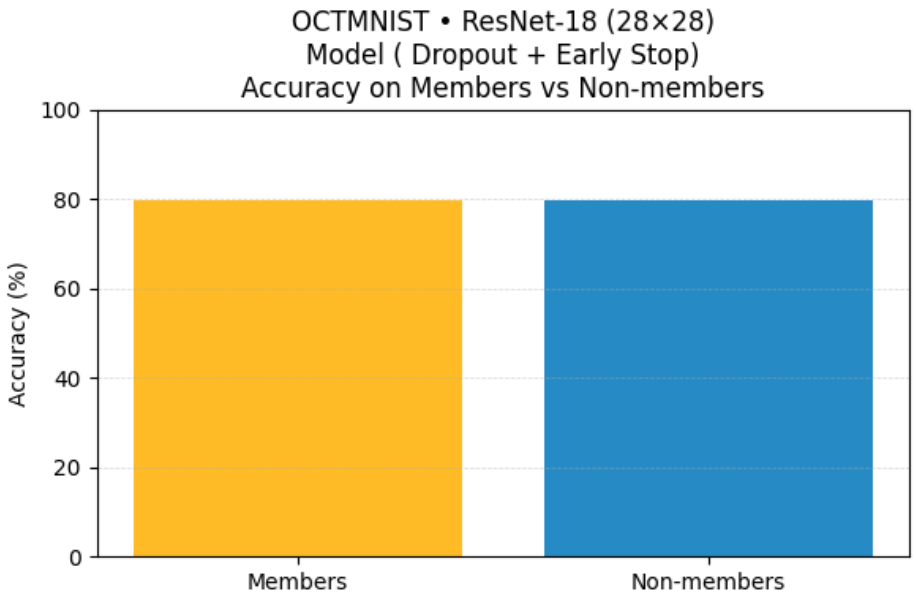
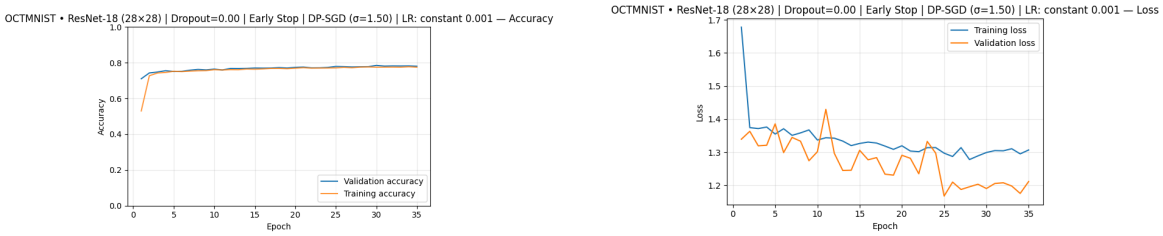


Figure A.136: Accuracy comparison for members and non-members (OCTMNIST + Regularisation with ε = 1.2).

Epsilon ε = 1.5



(a) Accuracy. **(b)** Loss.

Figure A.137: Training and validation performance for the OCTMNIST + Regularisation with ε = 1.5 .

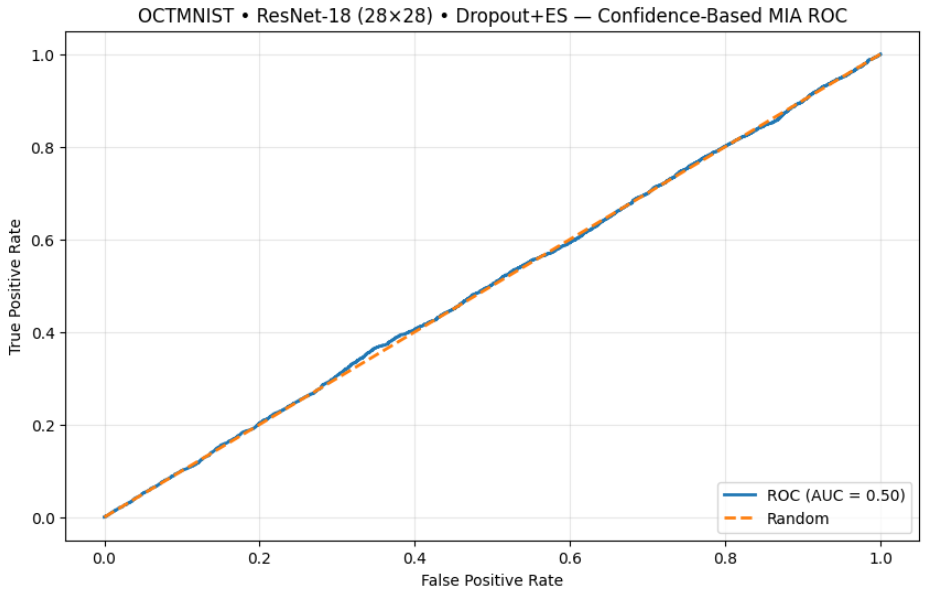


Figure A.138: ROC curve for confidence-based MIA (OCTMNIST + Regularisation with ε = 1.5).

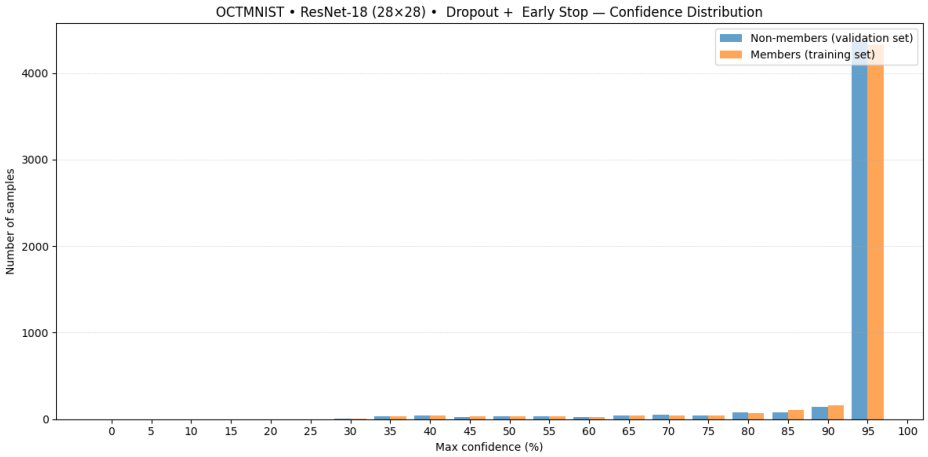


Figure A.139: Confidence distributions for members vs non-members (OCTMNIST + Regularisation with ε = 1.5).

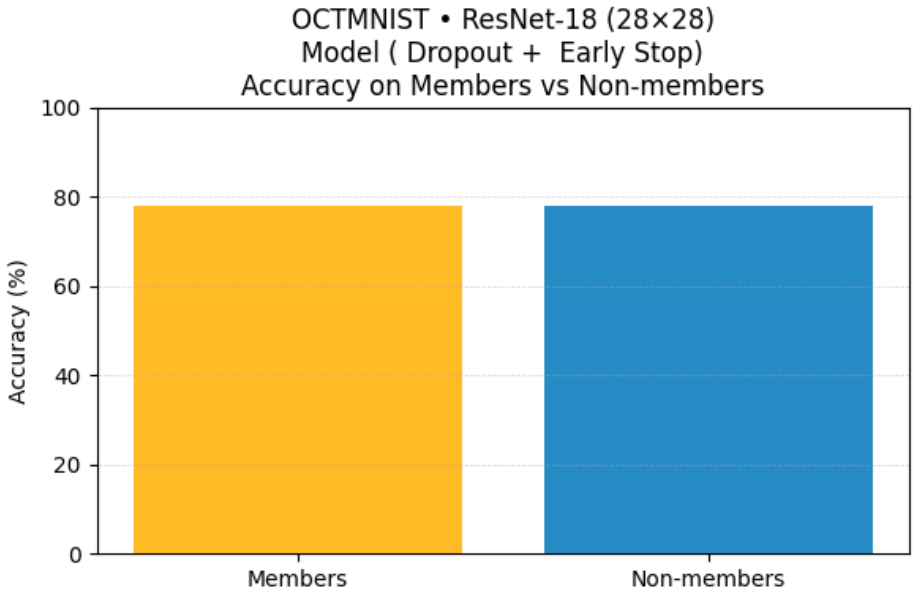
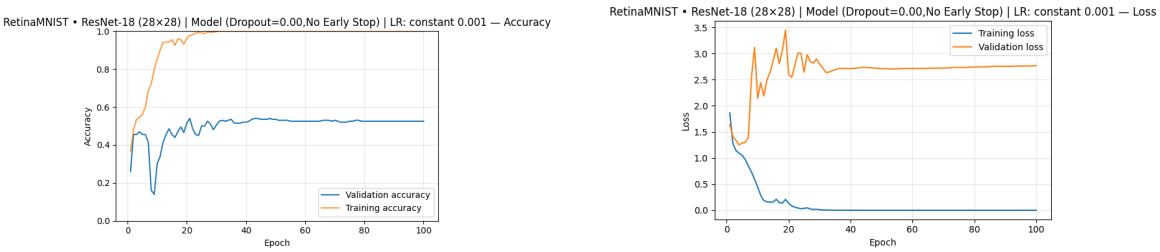


Figure A.140: Accuracy comparison for members and non-members (OCTMNIST + Regularisation with ε = 1.5).

A.4 RetinaMNIST

A.4.1 Baseline

LINK: `https://colab.research.google.com/drive/1AKu1WLEbgXAF6qtaver5BGk4rwD9xIn0?usp=sharing`



(a) Accuracy. **(b)** Loss.

Figure A.141: Training and validation performance for the RetinaMNIST baseline model.

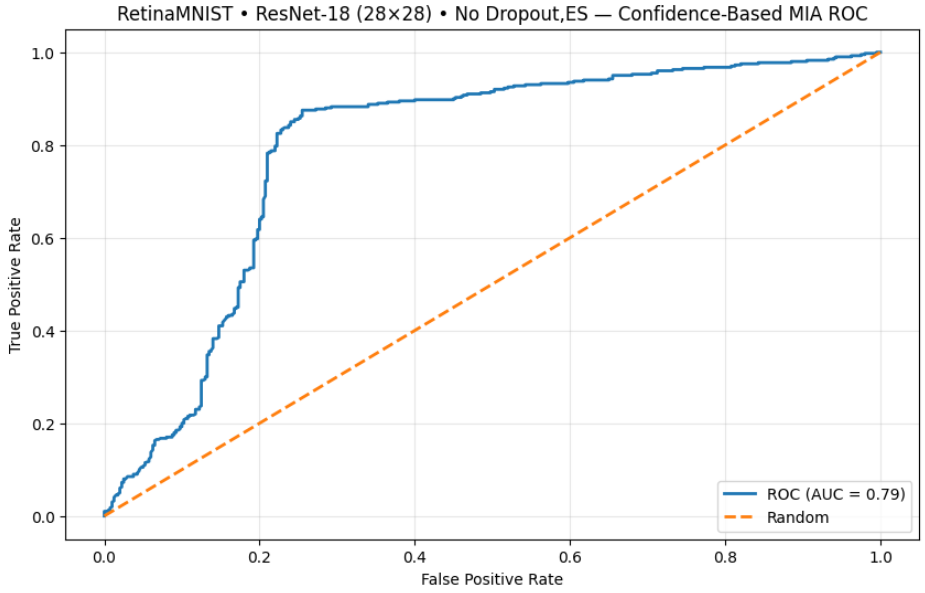


Figure A.142: ROC curve for confidence-based MIA (RetinaMNIST baseline).

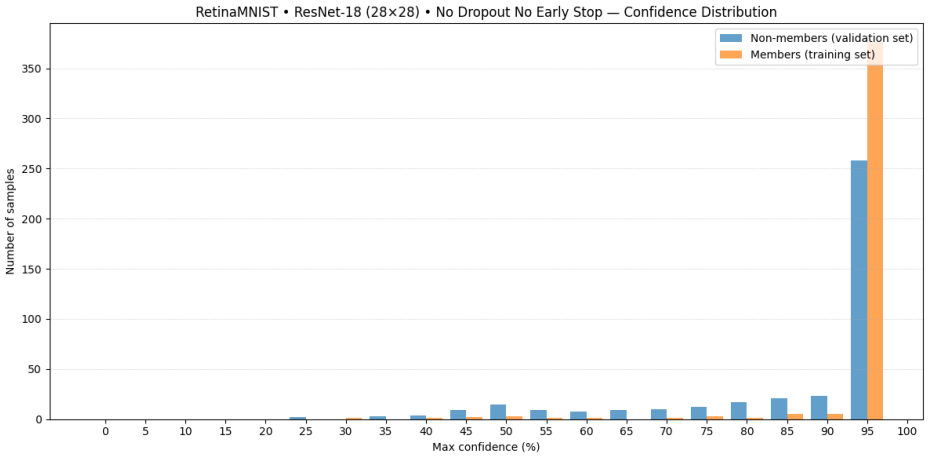


Figure A.143: Confidence distributions for members vs non-members (RetinaMNIST baseline).

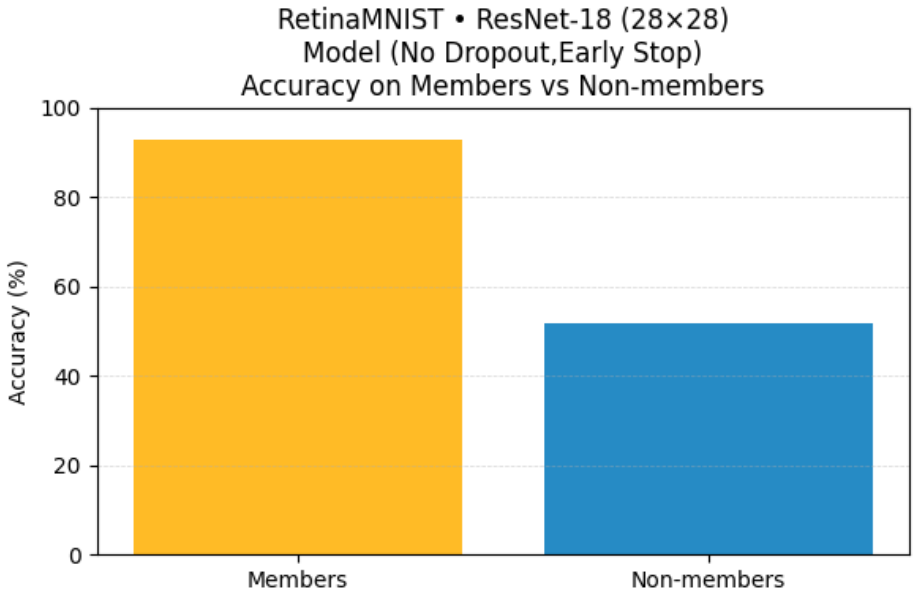
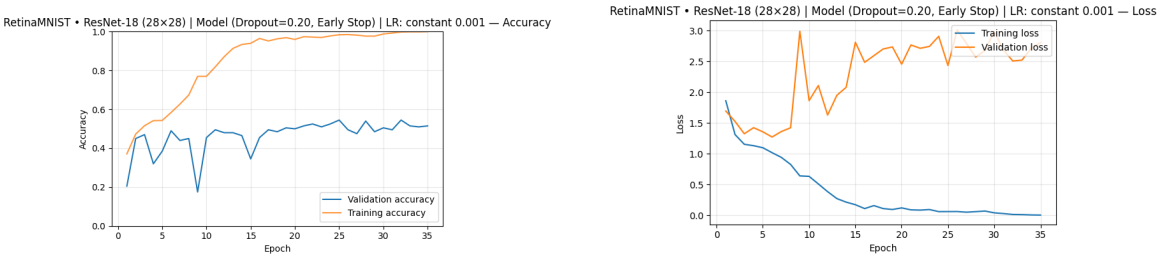


Figure A.144: Accuracy comparison for members and non-members (RetinaMNIST baseline).

A.4.2 Regularized



(a) Accuracy. **(b)** Loss.

Figure A.145: Training and validation performance for the RetinaMNIST Regularised model.

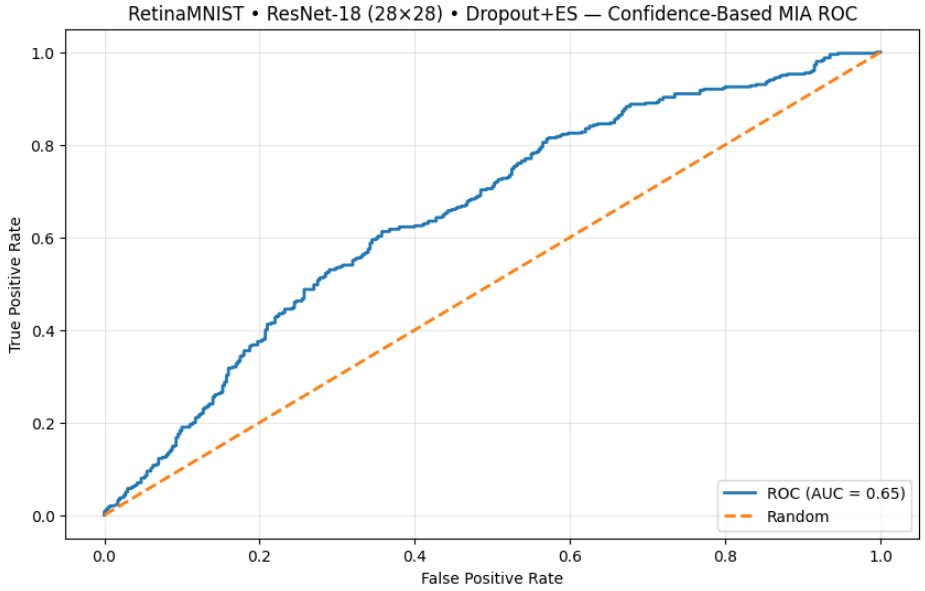


Figure A.146: ROC curve for confidence-based MIA (RetinaMNIST baseline).

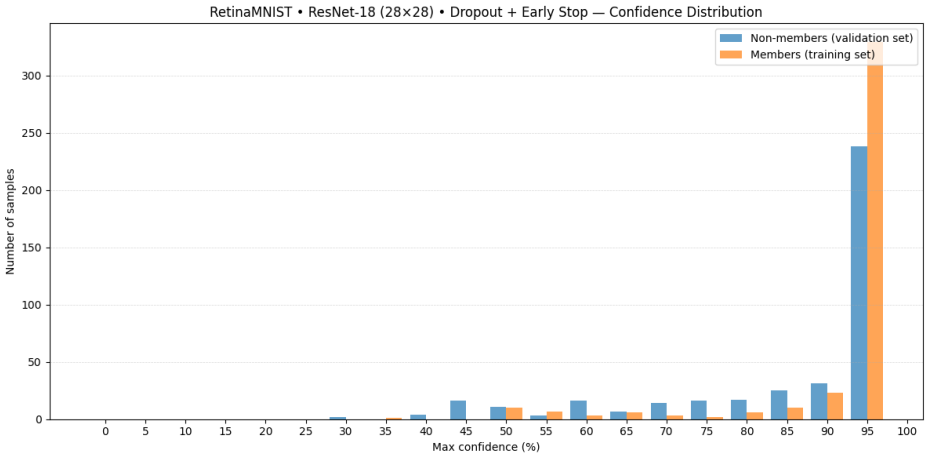


Figure A.147: Confidence distributions for members vs non-members (RetinaMNIST Regularised).

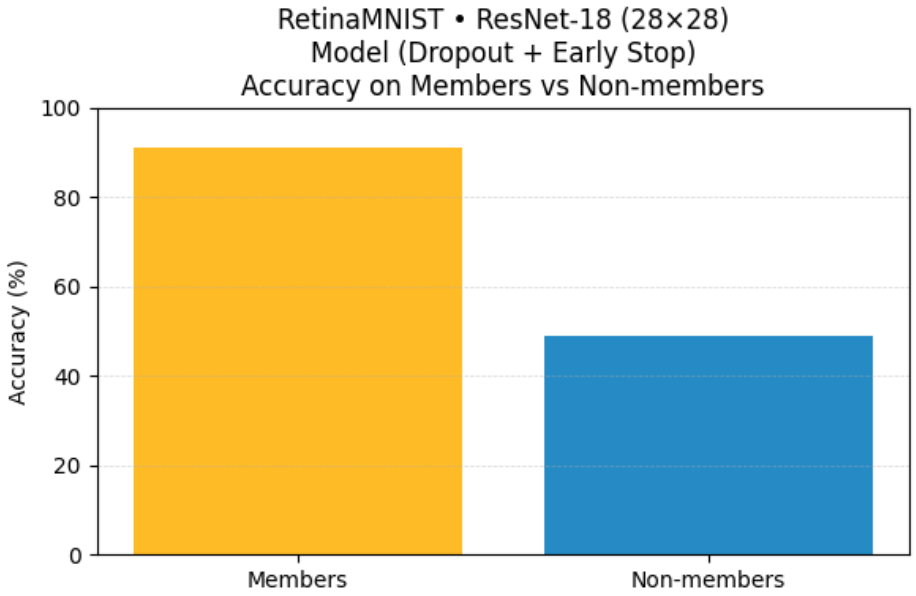
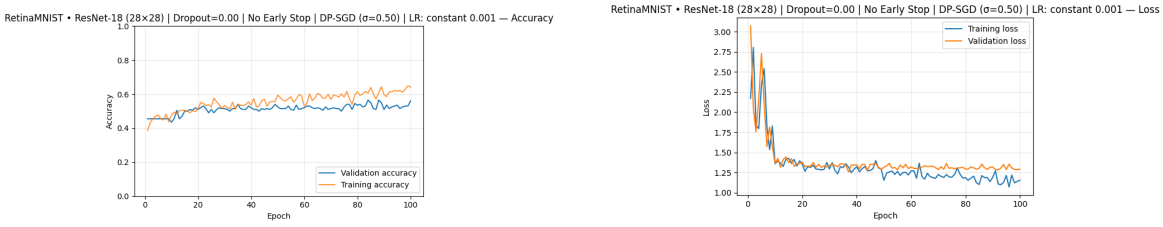


Figure A.148: Accuracy comparison for members and non-members (RetinaMNIST Regularised).

A.4.3 DP no Regularisation

LINK: `https://colab.research.google.com/drive/1IfY9CfeCMNvpU-dtn2RL9-VPpKxulrov?usp=sharing`

Epsilon ε = 0.5



(a) Accuracy. **(b)** Loss.

Figure A.149: Training and validation performance for the RetinaMNIST No Regularisation with ε = 0.5 .

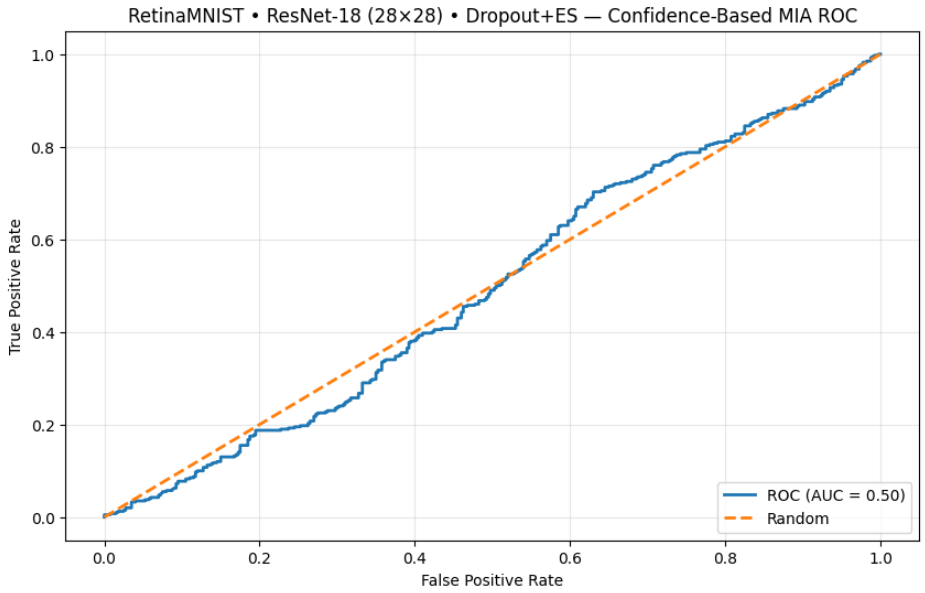


Figure A.150: ROC curve for confidence-based MIA (RetinaMNIST No Regularisation with ε = 0.5).

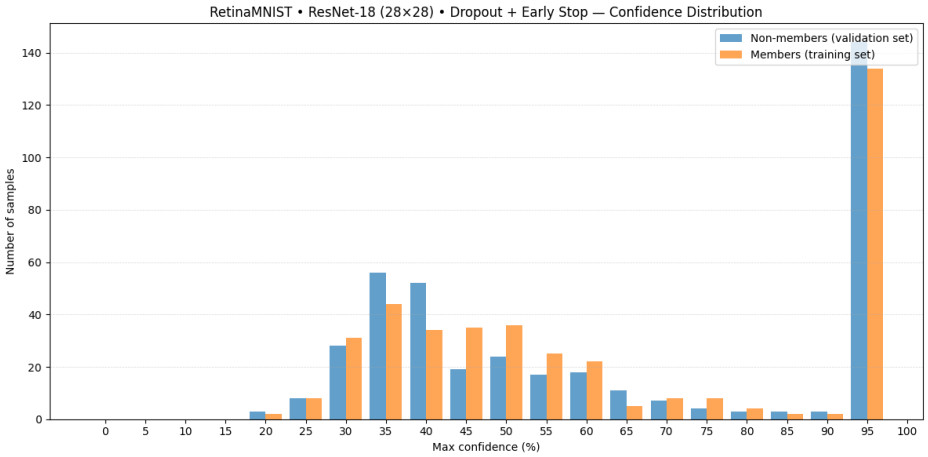


Figure A.151: Confidence distributions for members vs non-members (RetinaMNIST No Regularisation with $\varepsilon = 0.5$).

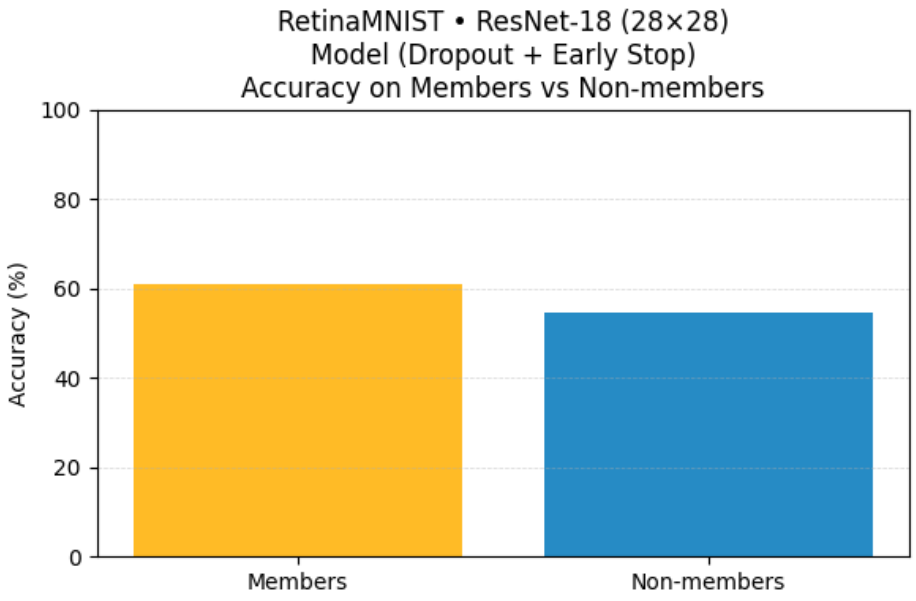
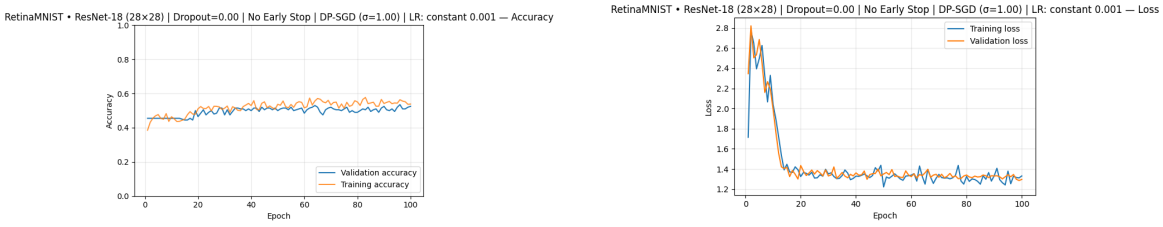


Figure A.152: Accuracy comparison for members and non-members (RetinaMNIST No Regularisation with $\varepsilon = 0.5$).

Epsilon ε = 1.0



(a) Accuracy. **(b)** Loss.

Figure A.153: Training and validation performance for the RetinaMNIST No Regularisation with ε = 1.0 .

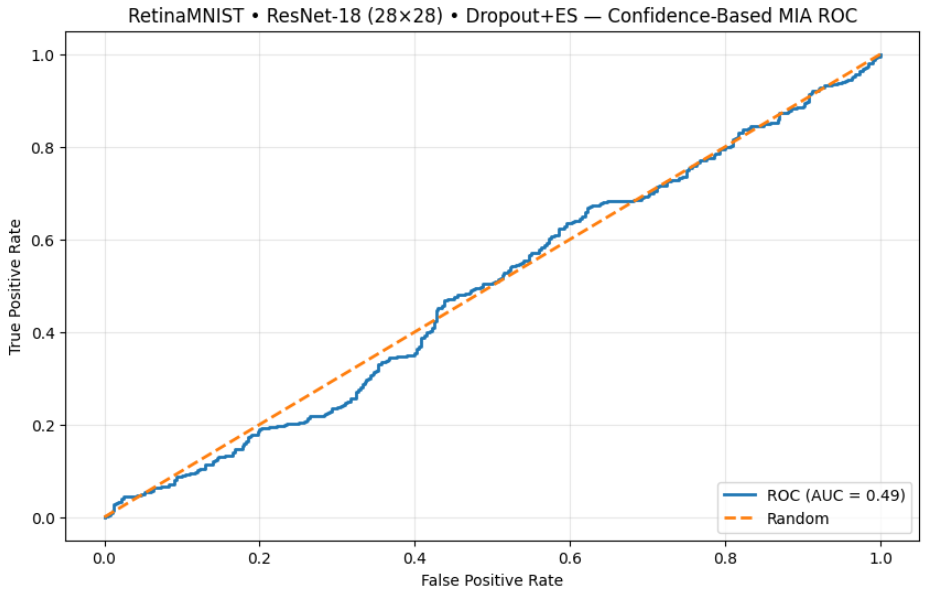


Figure A.154: ROC curve for confidence-based MIA (RetinaMNIST No Regularisation with ε = 1.0).

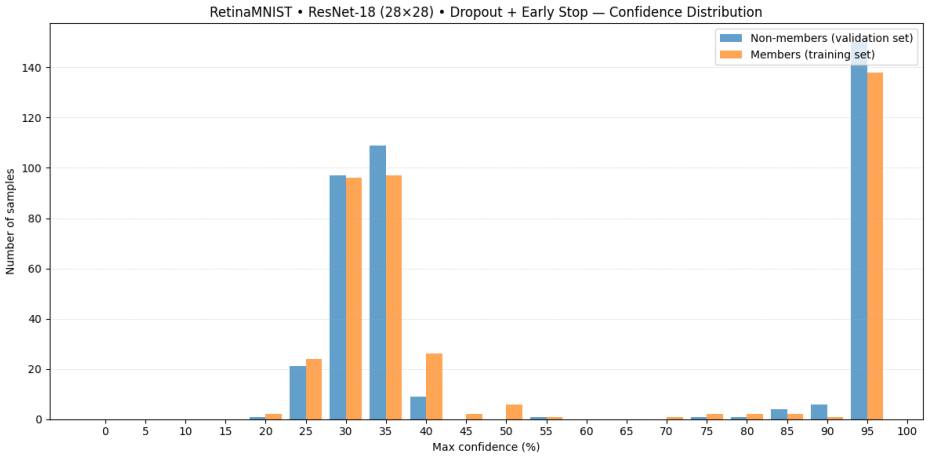


Figure A.155: Confidence distributions for members vs non-members (RetinaMNIST No Regularisation with ε = 1.0).

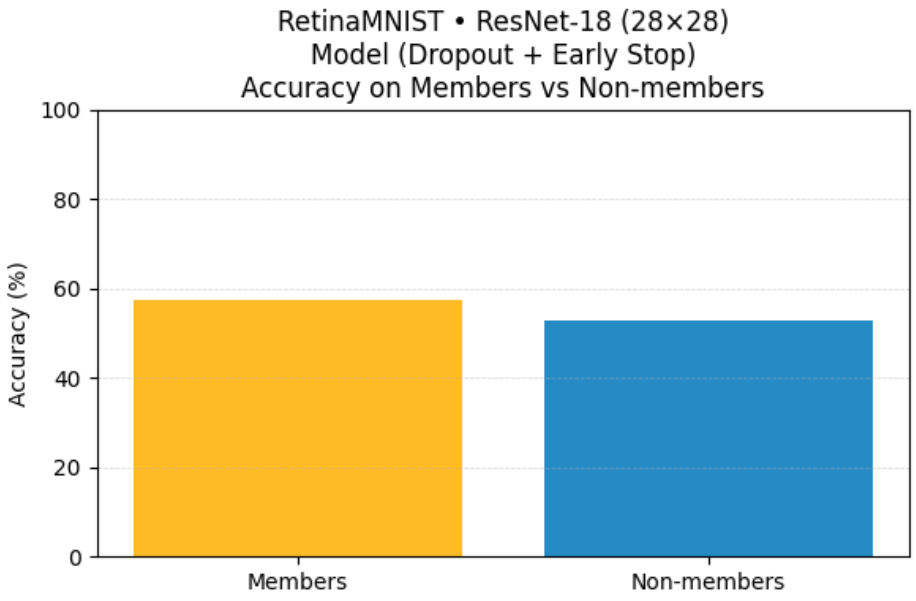
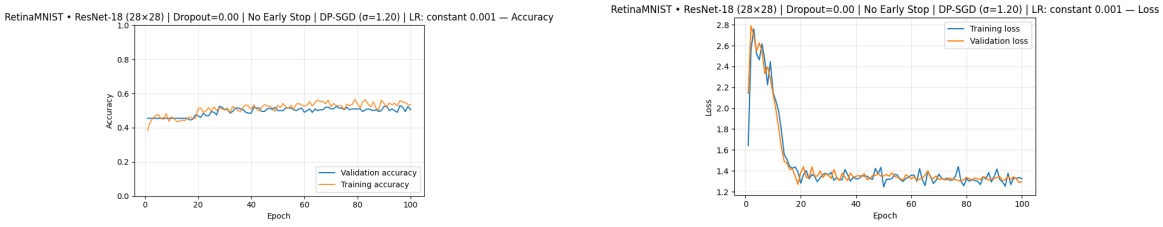


Figure A.156: Accuracy comparison for members and non-members (RetinaMNIST No Regularisation with ε = 1.0).

Epsilon ε = 1.2



(a) Accuracy. **(b)** Loss.

Figure A.157: Training and validation performance for the RetinaMNIST No Regularisation with ε = 1.2 .

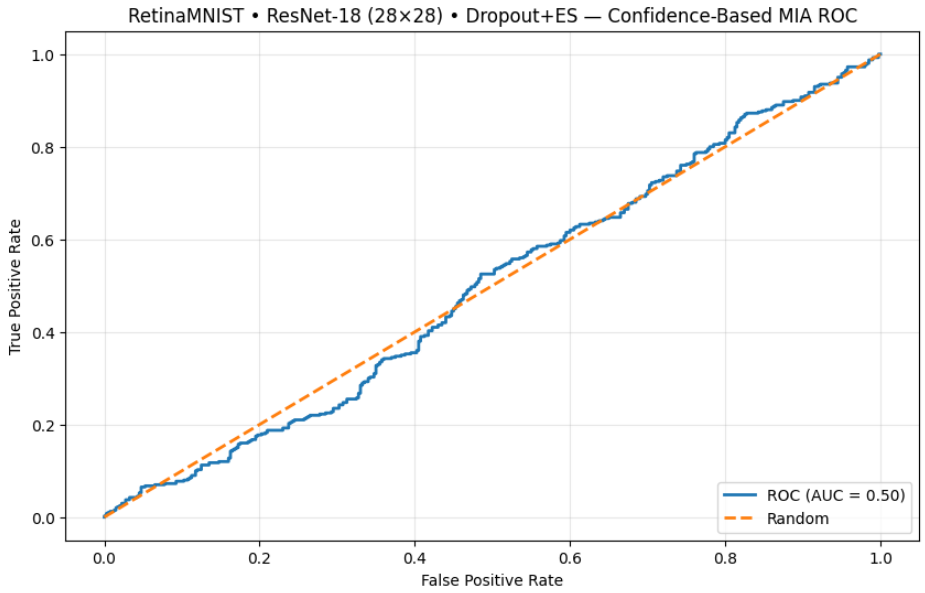


Figure A.158: ROC curve for confidence-based MIA (RetinaMNIST No Regularisation with ε = 1.2).

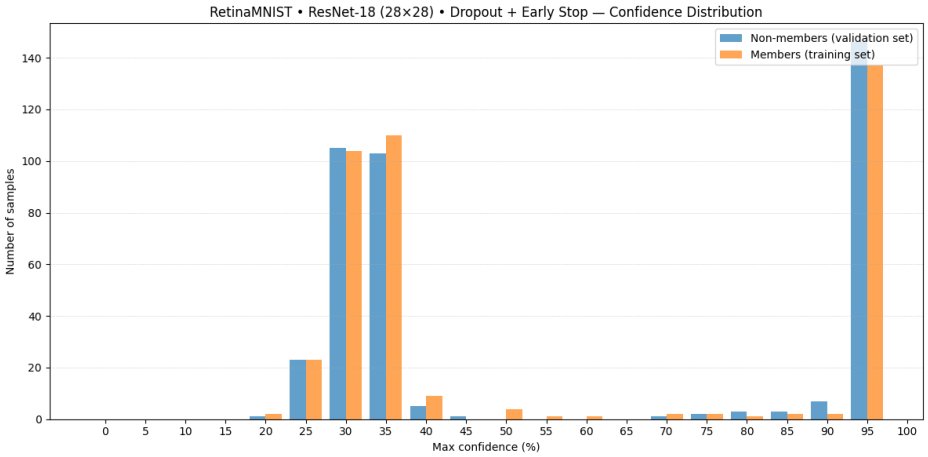


Figure A.159: Confidence distributions for members vs non-members (RetinaMNIST No Regularisation with $\varepsilon = 1.2$).

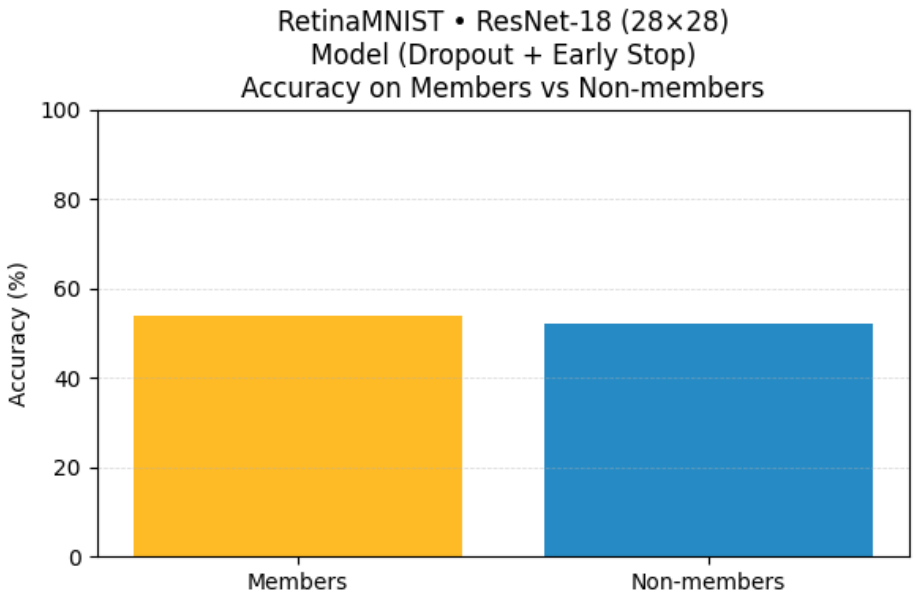
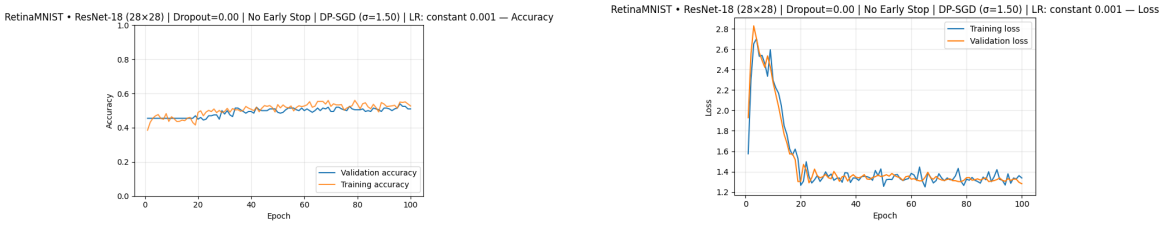


Figure A.160: Accuracy comparison for members and non-members (RetinaMNIST No Regularisation with $\varepsilon = 1.2$).

Epsilon ε = 1.5



(a) Accuracy. **(b)** Loss.

Figure A.161: Training and validation performance for the RetinaMNIST No Regularisation with ε = 1.5 .

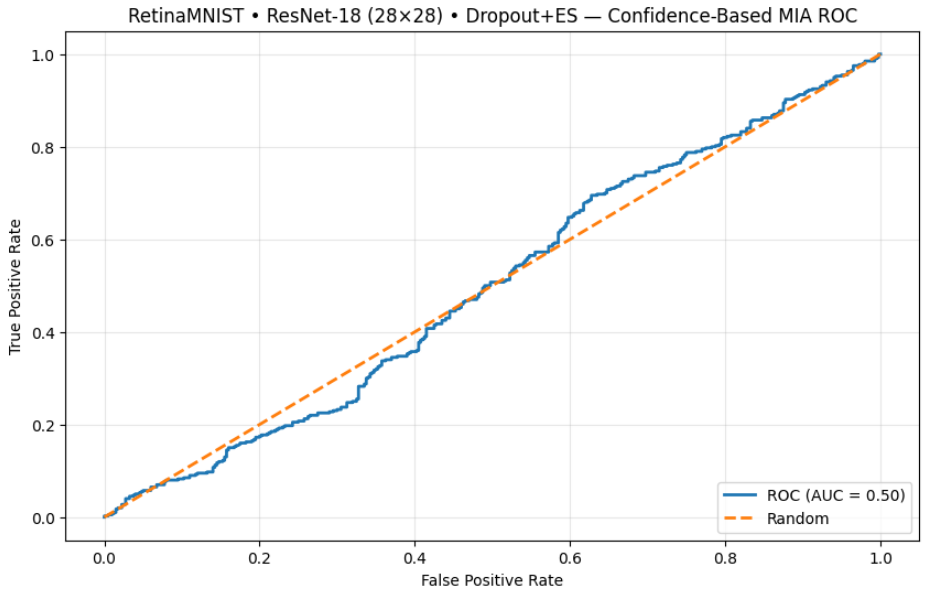


Figure A.162: ROC curve for confidence-based MIA (RetinaMNIST No Regularisation with ε = 1.5).

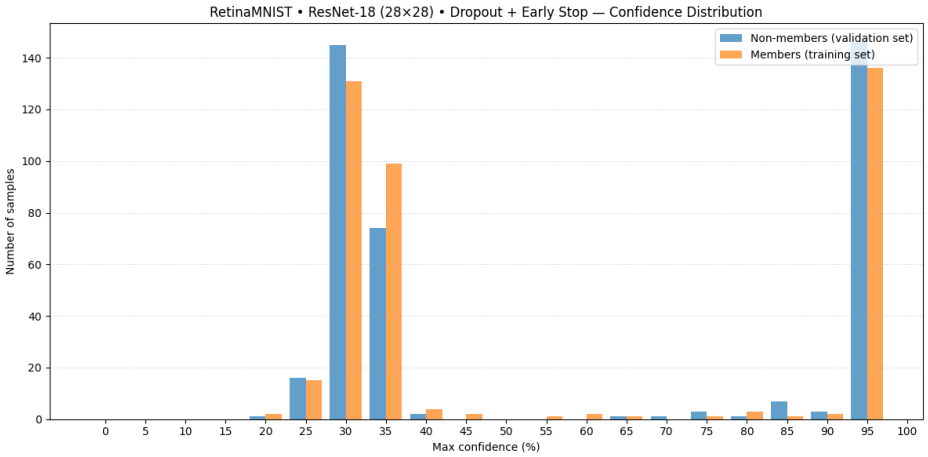


Figure A.163: Confidence distributions for members vs non-members (RetinaMNIST No Regularisation with $\varepsilon = 1.5$).

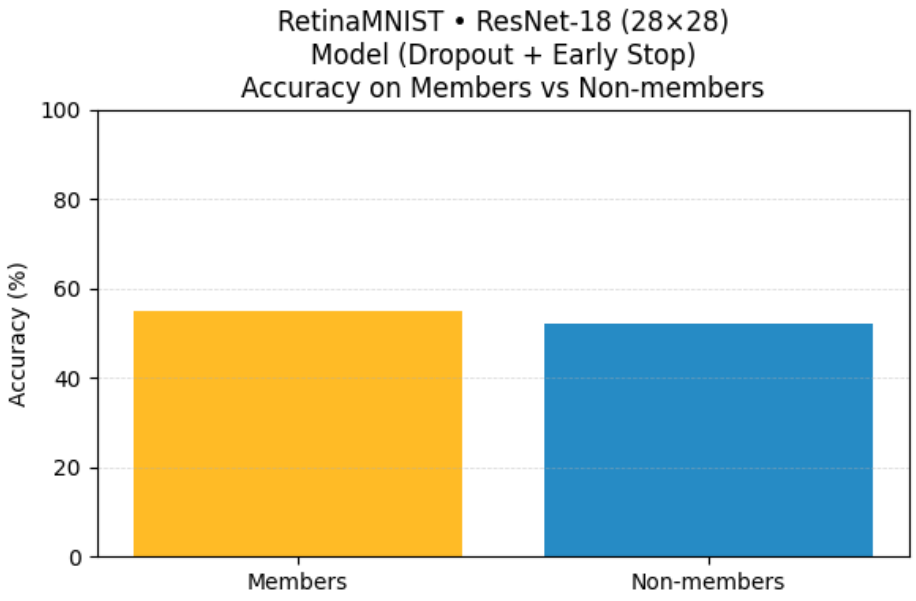
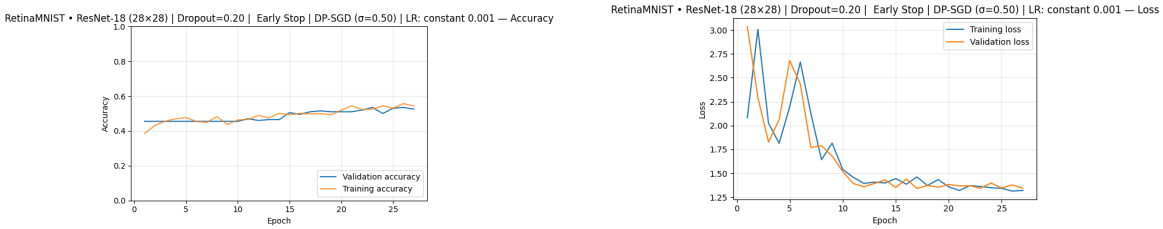


Figure A.164: Accuracy comparison for members and non-members (RetinaMNIST No Regularisation with $\varepsilon = 1.5$).

A.4.4 DP + Regularisation

LINK: `https://colab.research.google.com/drive/1OCB9RCStKs21NqFFX50VEKktmiW9dC54?usp=sharing`

Epsilon ε = 0.5



(a) Accuracy. **(b)** Loss.

Figure A.165: Training and validation performance for the RetinaMNIST + Regularisation with ε = 0.5 .

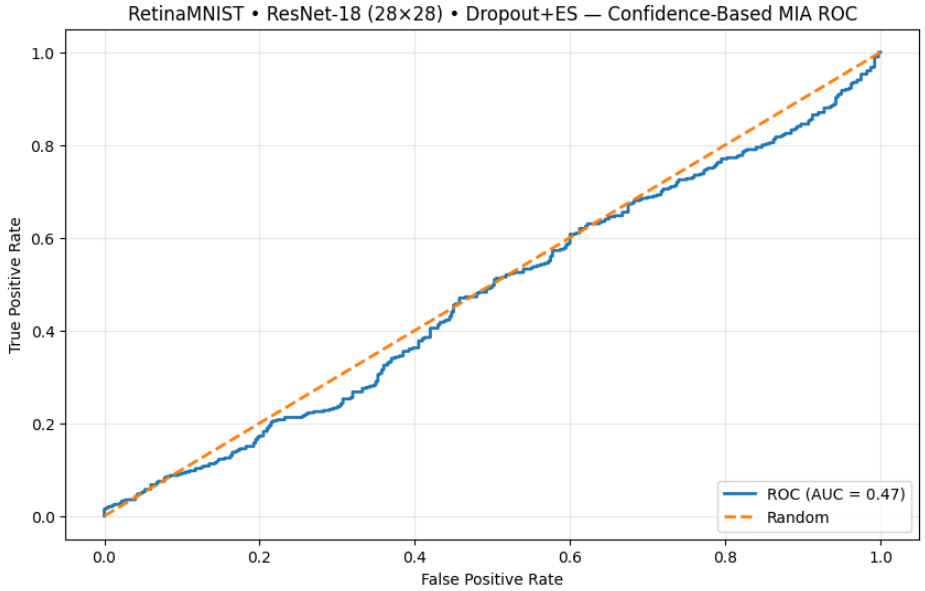


Figure A.166: ROC curve for confidence-based MIA (RetinaMNIST + Regularisation with ε = 0.5).

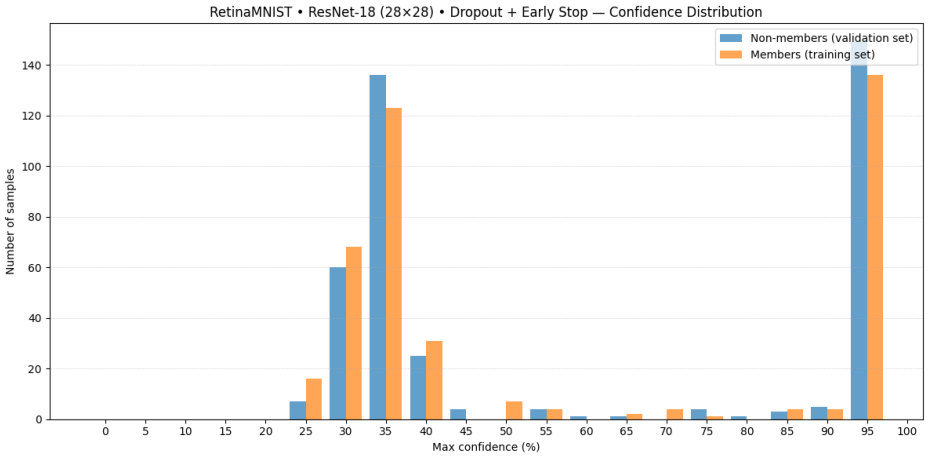


Figure A.167: Confidence distributions for members vs non-members (RetinaMNIST + Regularisation with ε = 0.5).

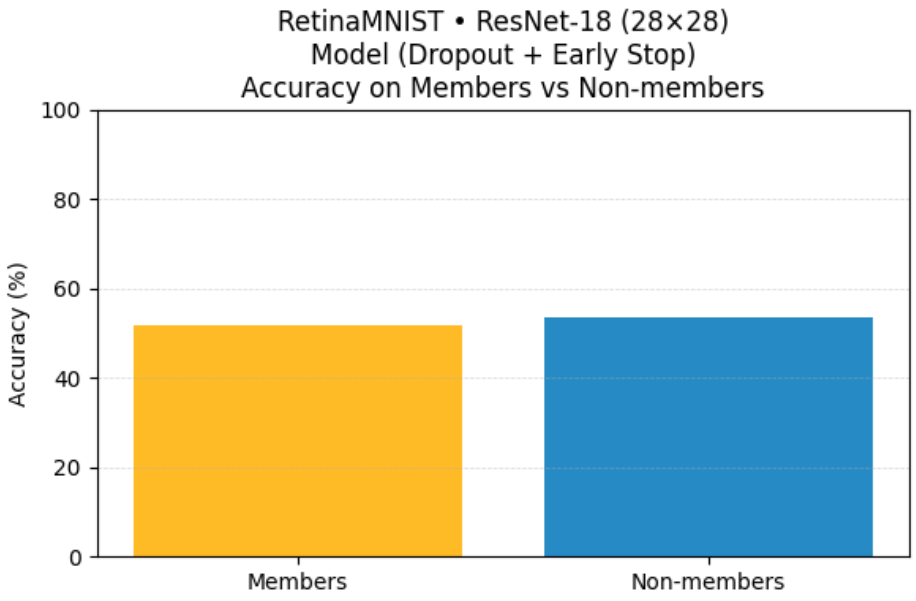
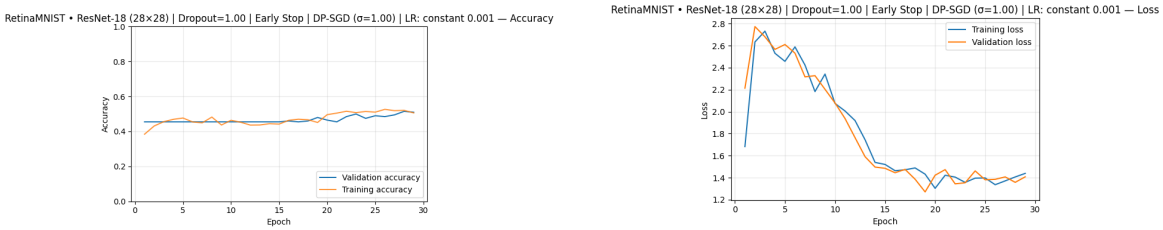


Figure A.168: Accuracy comparison for members and non-members (RetinaMNIST + Regularisation with ε = 0.5).

Epsilon ε = 1.0



(a) Accuracy. **(b)** Loss.

Figure A.169: Training and validation performance for the RetinaMNIST + Regularisation with ε = 1.0 .

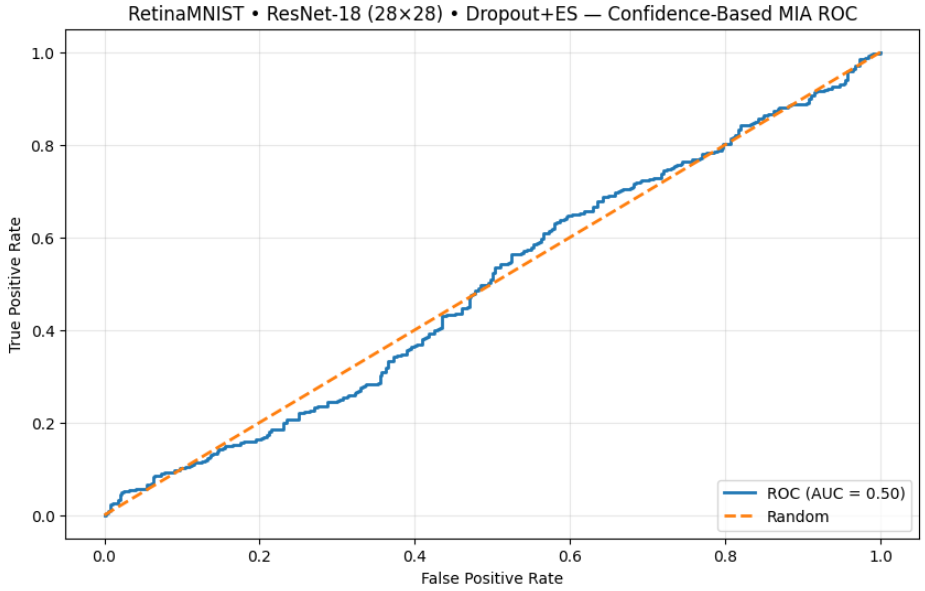


Figure A.170: ROC curve for confidence-based MIA (RetinaMNIST + Regularisation with ε = 1.0).

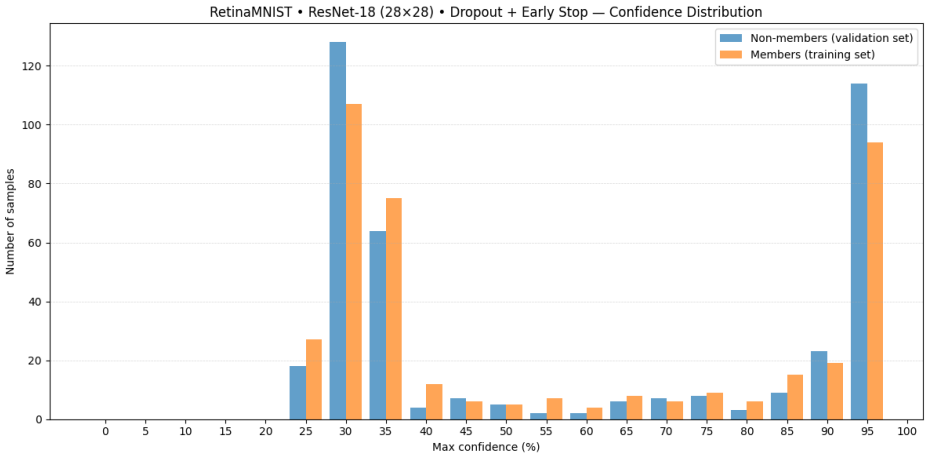


Figure A.171: Confidence distributions for members vs non-members (RetinaMNIST + Regularisation with ε = 1.0).

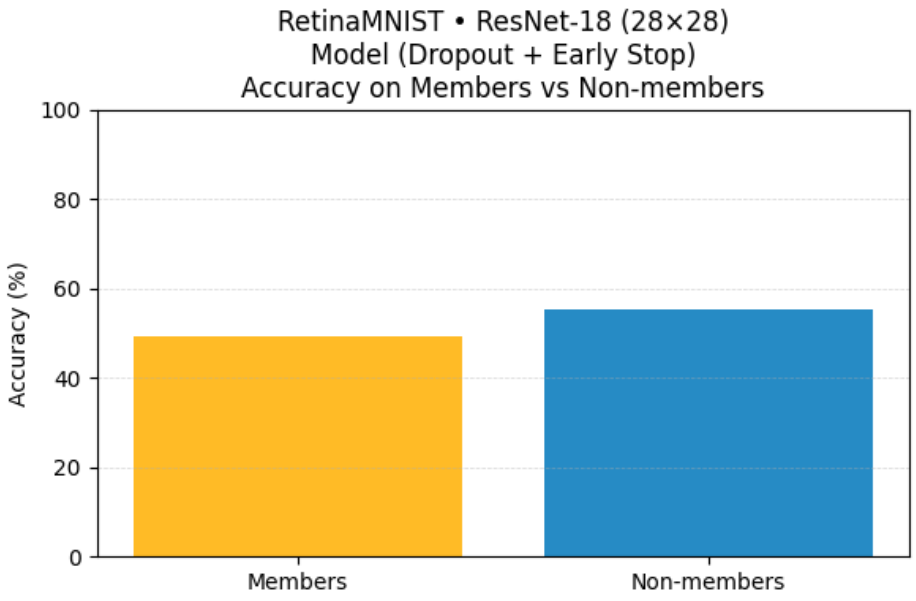
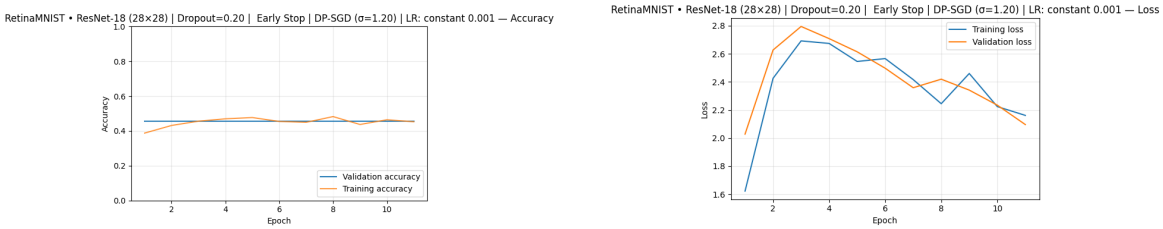


Figure A.172: Accuracy comparison for members and non-members (RetinaMNIST + Regularisation with ε = 1.0).

Epsilon ε = 1.2



(a) Accuracy. **(b)** Loss.

Figure A.173: Training and validation performance for the RetinaMNIST + Regularisation with ε = 1.2 .

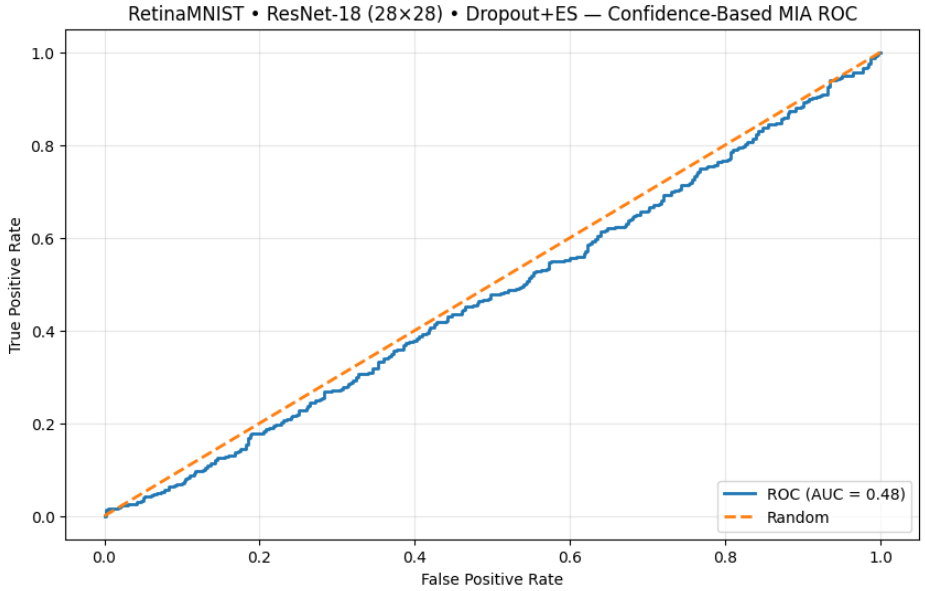


Figure A.174: ROC curve for confidence-based MIA (RetinaMNIST + Regularisation with ε = 1.2).

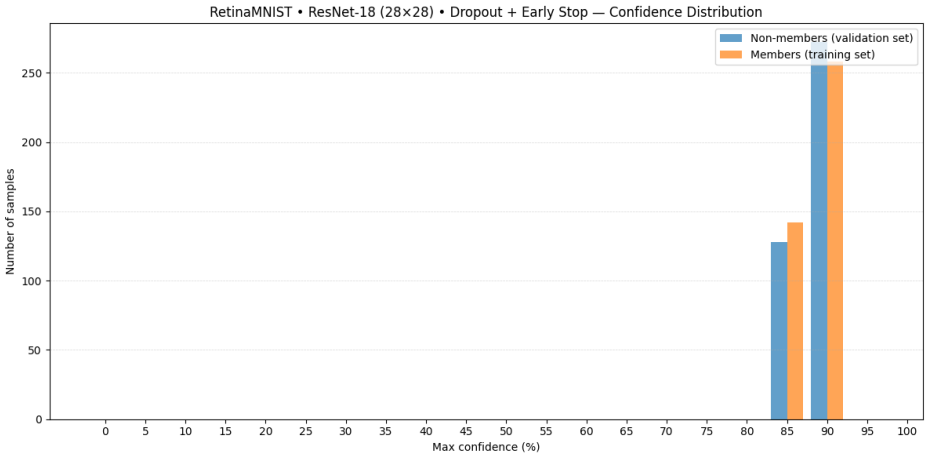


Figure A.175: Confidence distributions for members vs non-members (RetinaMNIST + Regularisation with ε = 1.2).

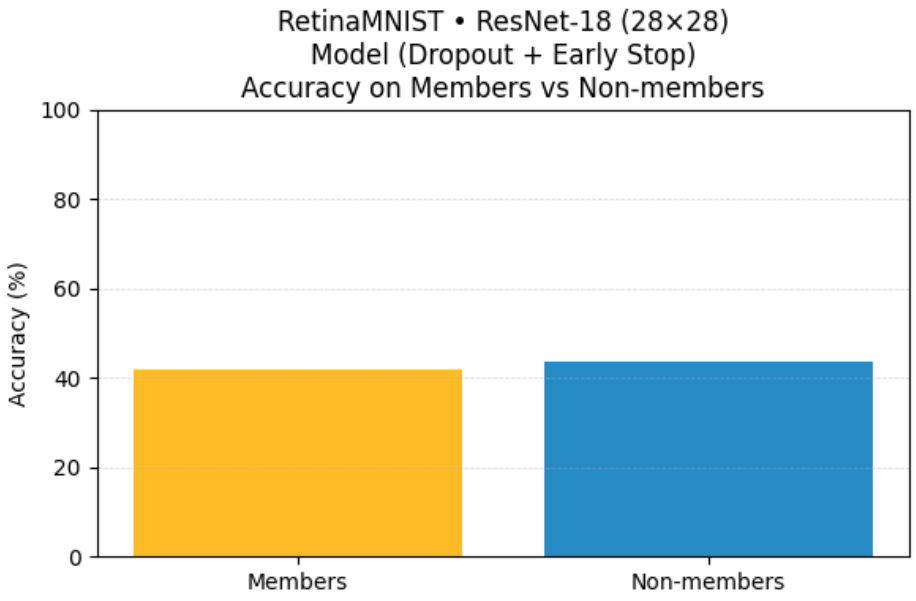
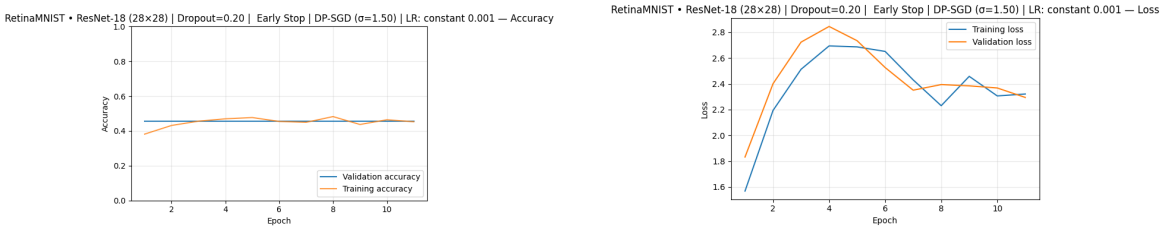


Figure A.176: Accuracy comparison for members and non-members (RetinaMNIST + Regularisation with ε = 1.2).

Epsilon ε = 1.5



(a) Accuracy. **(b)** Loss.

Figure A.177: Training and validation performance for the RetinaMNIST + Regularisation with ε = 1.5 .

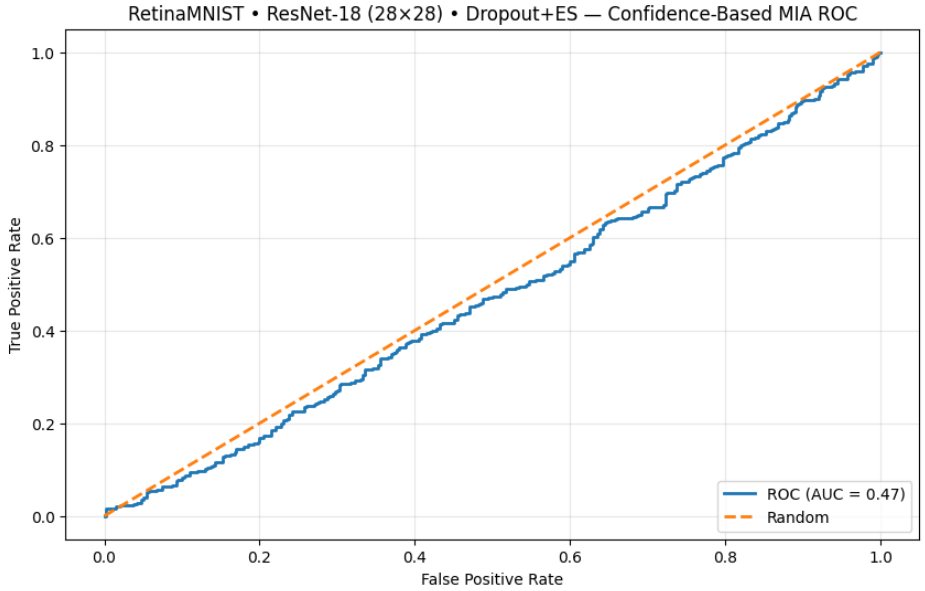


Figure A.178: ROC curve for confidence-based MIA (RetinaMNIST + Regularisation with ε = 1.5).

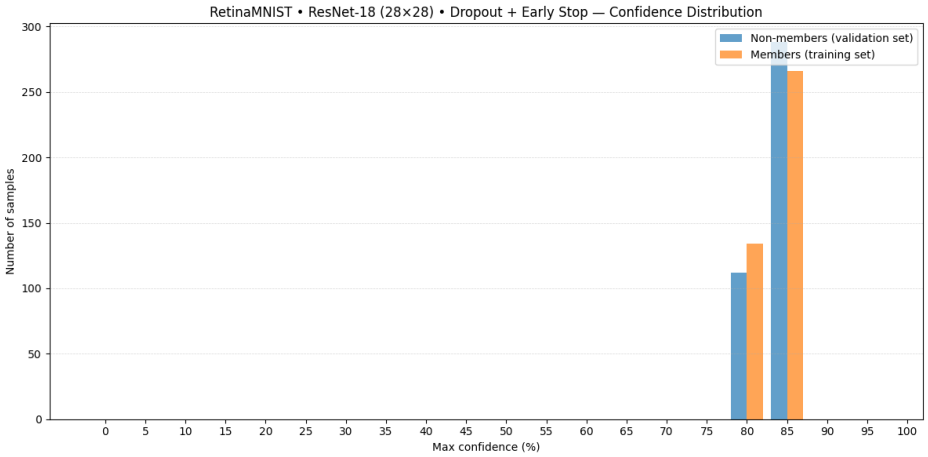


Figure A.179: Confidence distributions for members vs non-members (RetinaMNIST + Regularisation with ε = 1.5).

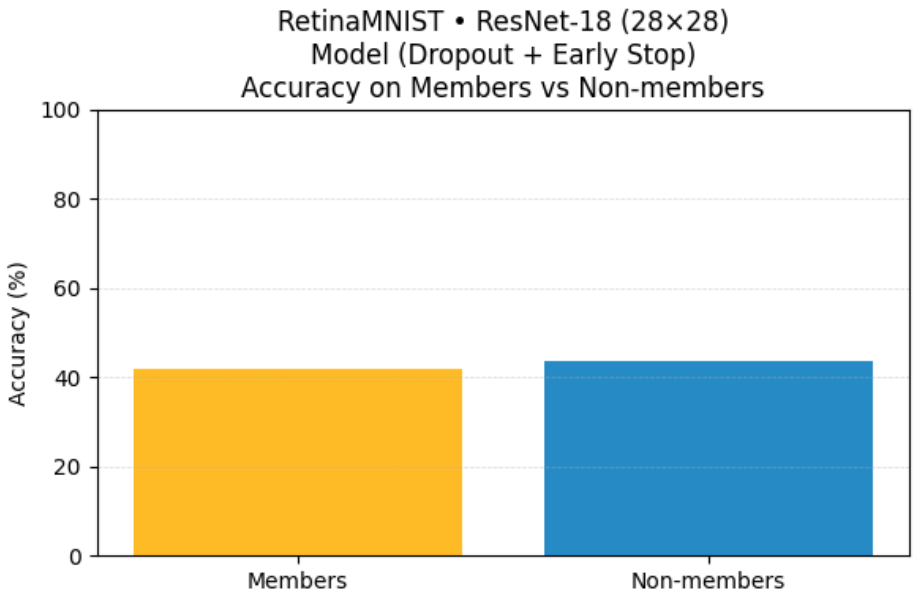
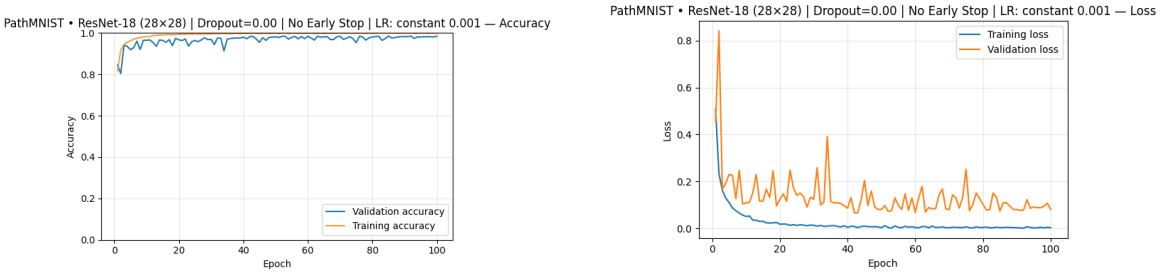


Figure A.180: Accuracy comparison for members and non-members (RetinaMNIST + Regularisation with ε = 1.5).

A.5 PathMNIST

A.5.1 Baseline

LINK: `https://colab.research.google.com/drive/1sMi8ViLR5g-p6s-Hrdfm-Ec3QEYT1SqY?usp=sharing`



(a) Accuracy. **(b)** Loss.

Figure A.181: Training and validation performance for the PathMNIST baseline model.

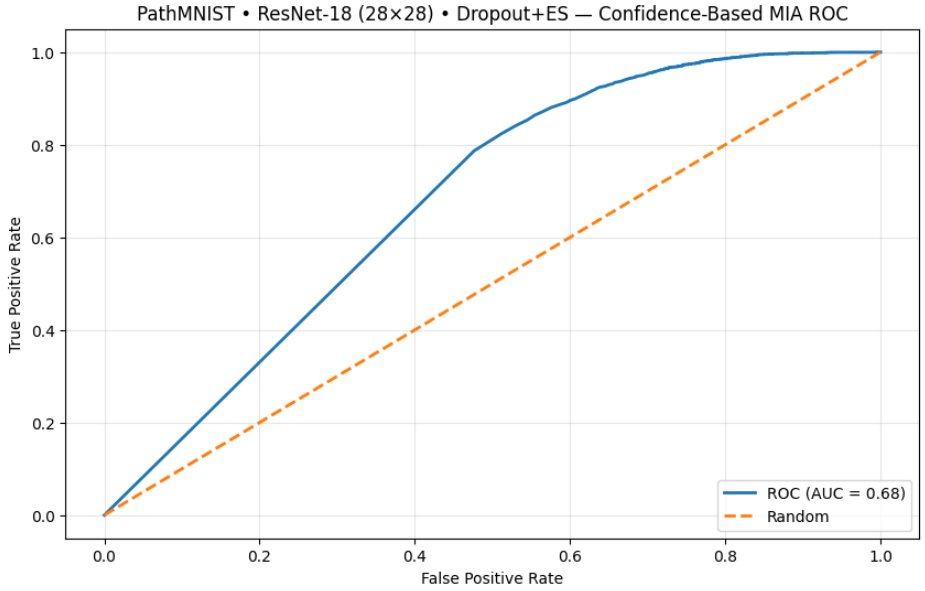


Figure A.182: ROC curve for confidence-based MIA (PathMNIST baseline).

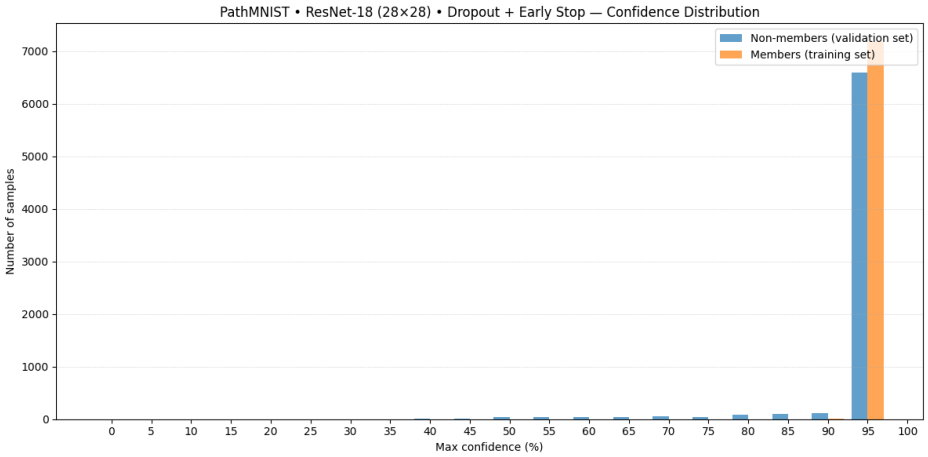


Figure A.183: Confidence distributions for members vs non-members (PathMNIST baseline).

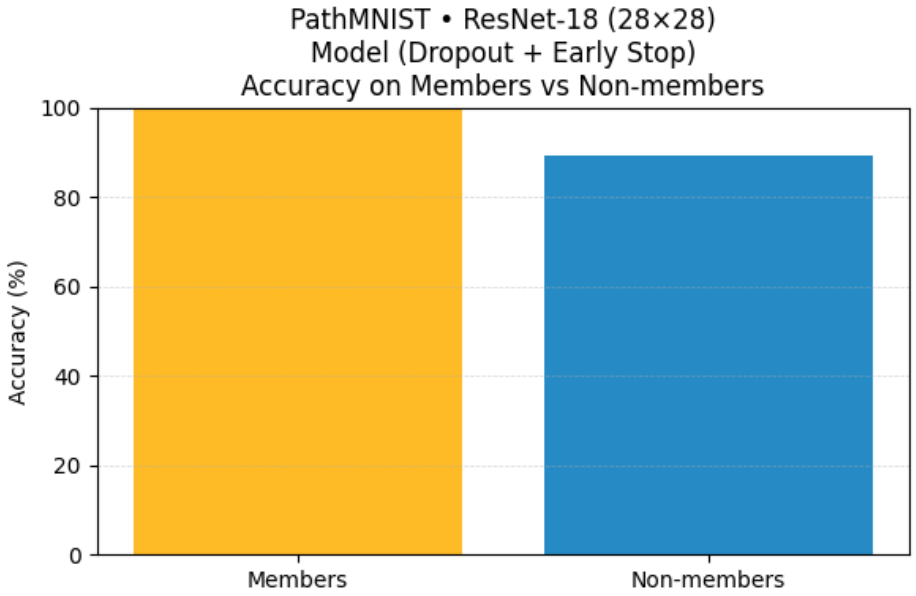
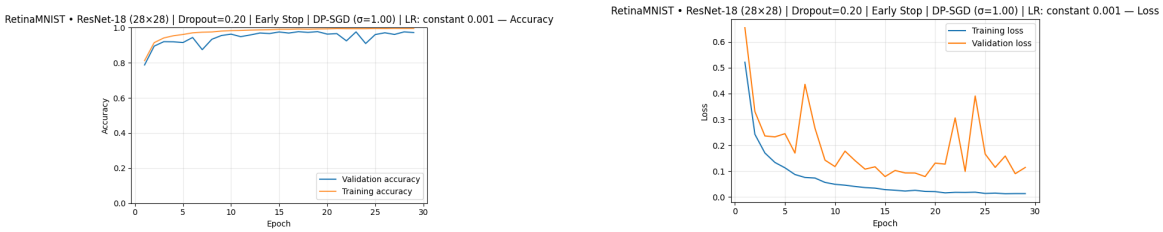


Figure A.184: Accuracy comparison for members and non-members (PathMNIST baseline).

A.5.2 Regularised

LINK: `https://colab.research.google.com/drive/1IvHNmswCjTmmdCMDcM0Aden59xXY7xGM?usp=sharing`



(a) Accuracy. **(b)** Loss.

Figure A.185: Training and validation performance for the PathMNIST Regularised) model.

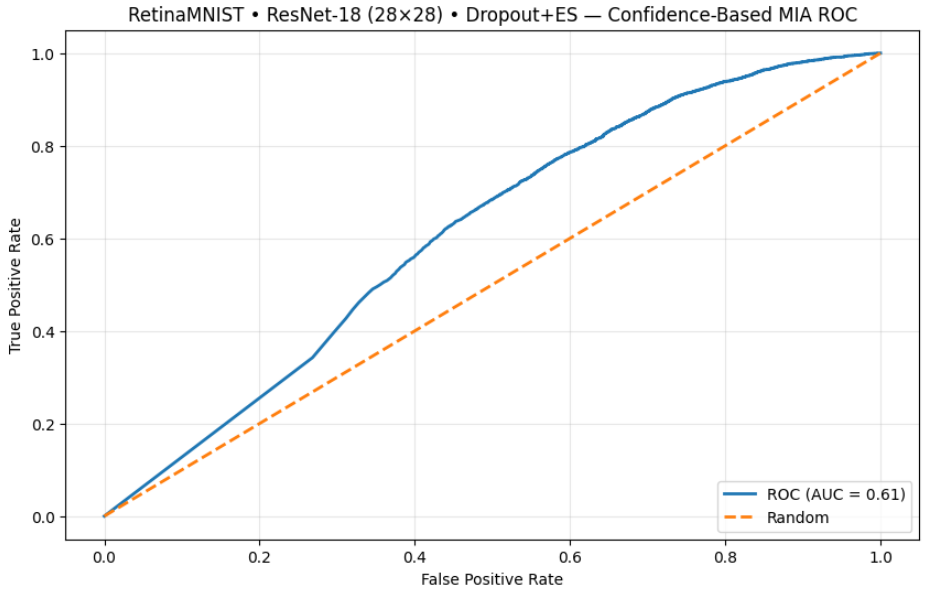


Figure A.186: ROC curve for confidence-based MIA (PathMNIST Regularised).

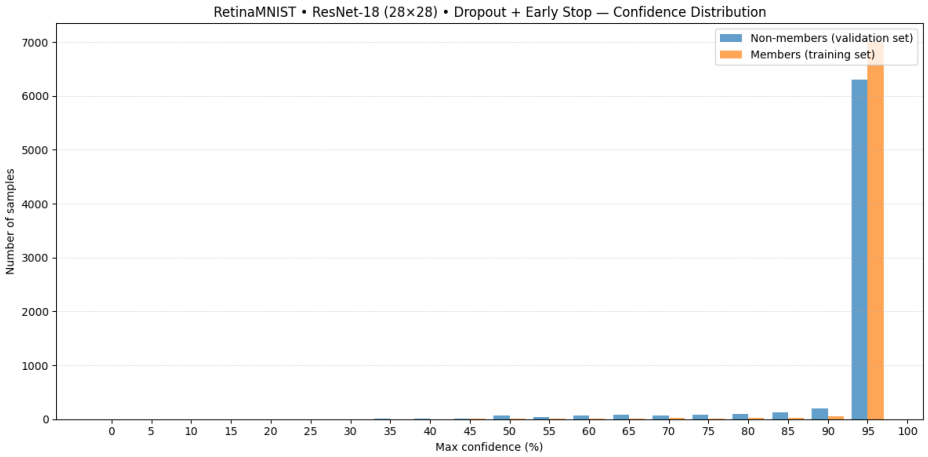


Figure A.187: Confidence distributions for members vs non-members (PathMNIST Regularised)).

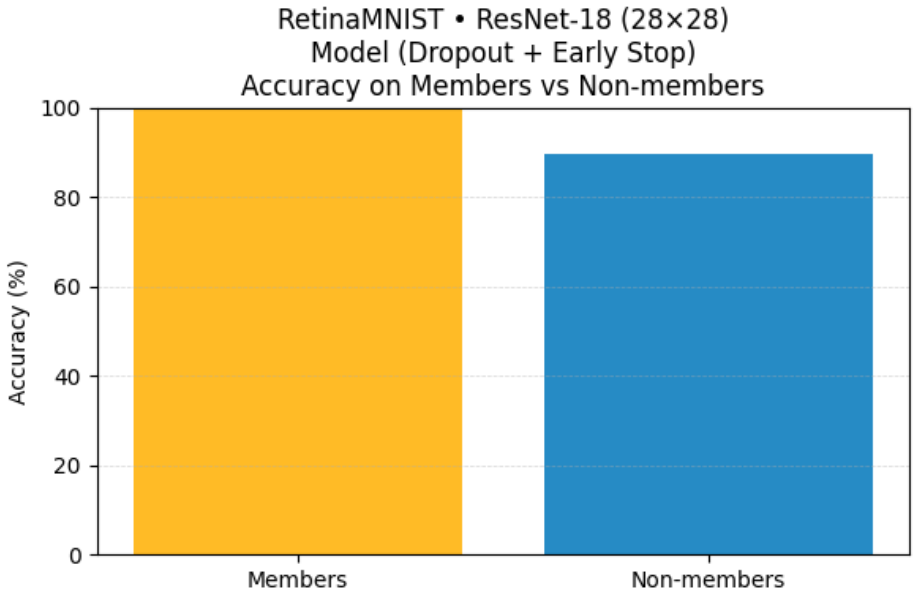
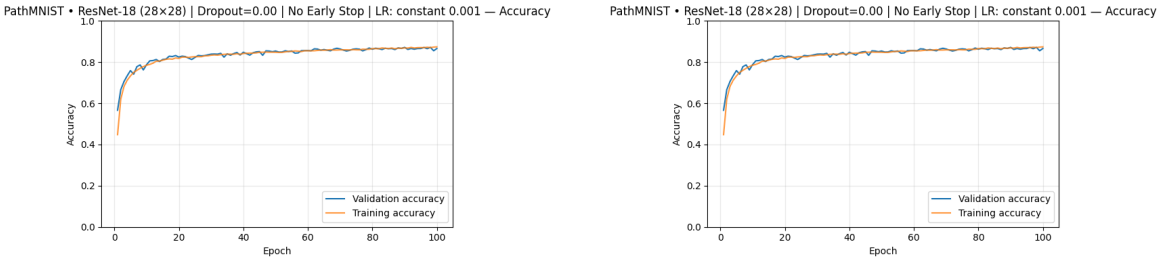


Figure A.188: Accuracy comparison for members and non-members (PathMNIST Regularised)).

A.5.3 DP no Regularisation

LINK: `https://colab.research.google.com/drive/12TvBtby07iEKBsFLQuKRoqO1MQvzV6TO?usp=sharing`

ε = 0.5



(a) Accuracy. **(b)** Loss.

Figure A.189: Training and validation performance for the PathMNIST No Regularisation with ε = 0.5 .

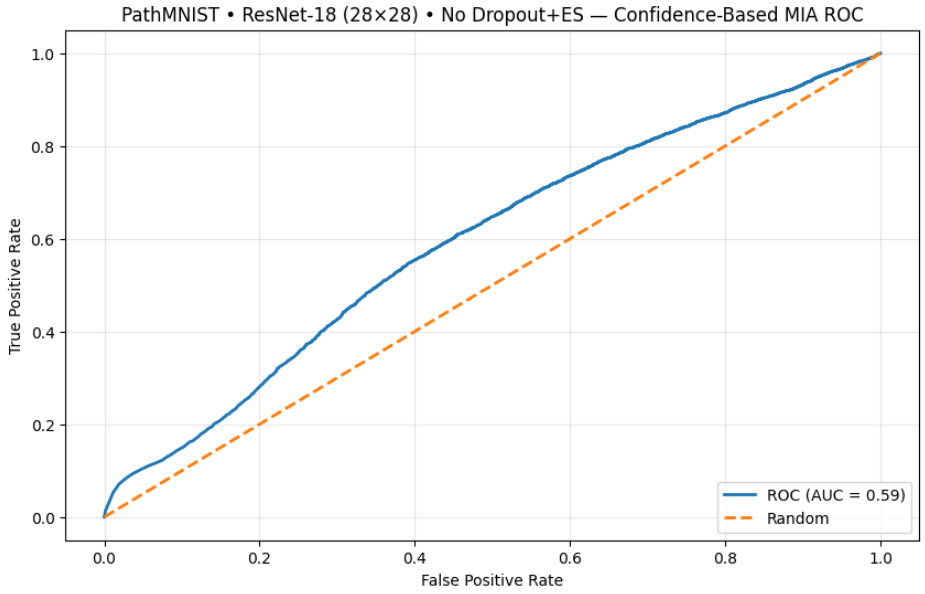


Figure A.190: ROC curve for confidence-based MIA (PathMNIST No Regularisation with ε = 0.5).

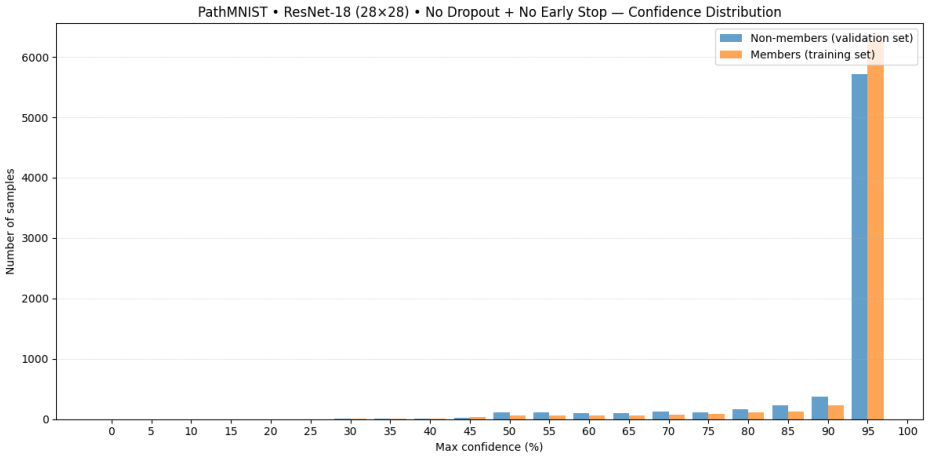


Figure A.191: Confidence distributions for members vs non-members (PathMNIST No Regularisation with ε = 0.5).

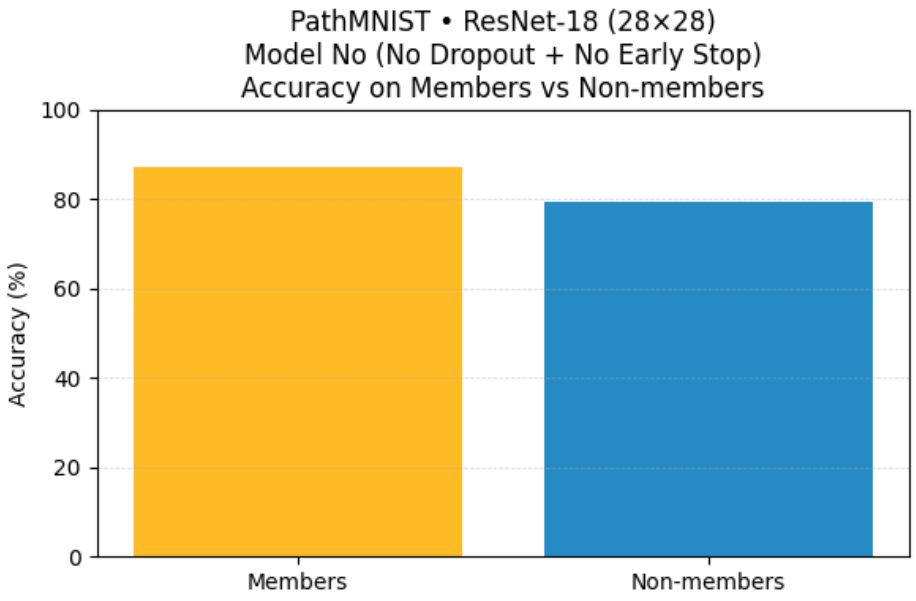
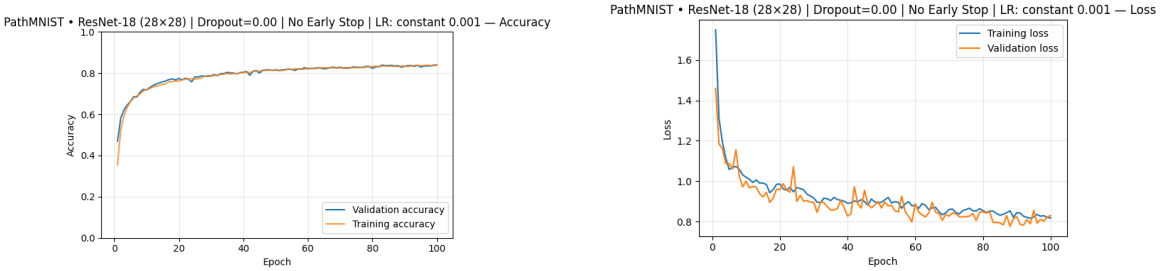


Figure A.192: Accuracy comparison for members and non-members (PathMNIST No Regularisation with ε = 0.5).

ε = 1.0



(a) Accuracy. **(b)** Loss.

Figure A.193: Training and validation performance for the PathMNIST No Regularisation with ε = 1.0 .

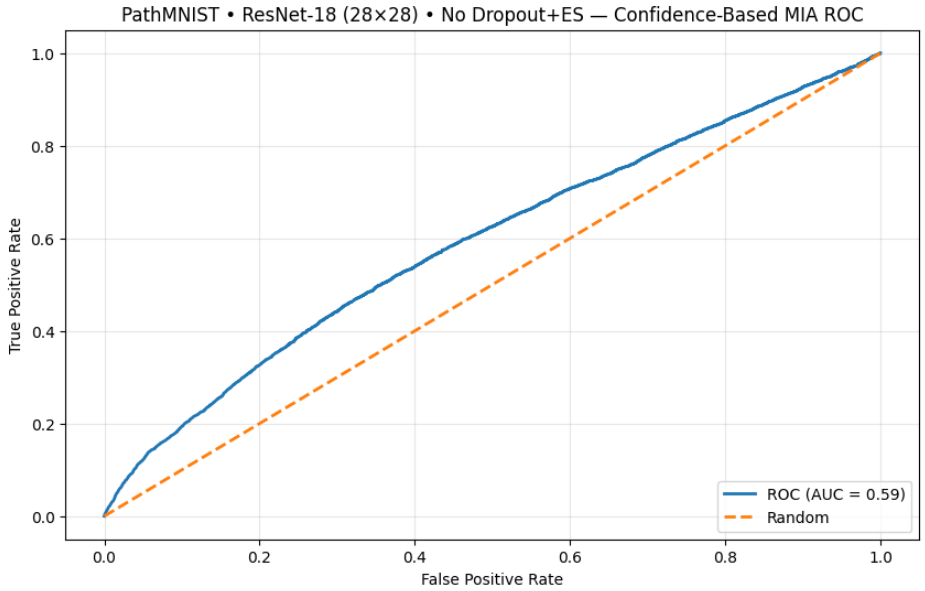


Figure A.194: ROC curve for confidence-based MIA (PathMNIST No Regularisation with ε = 1.0).

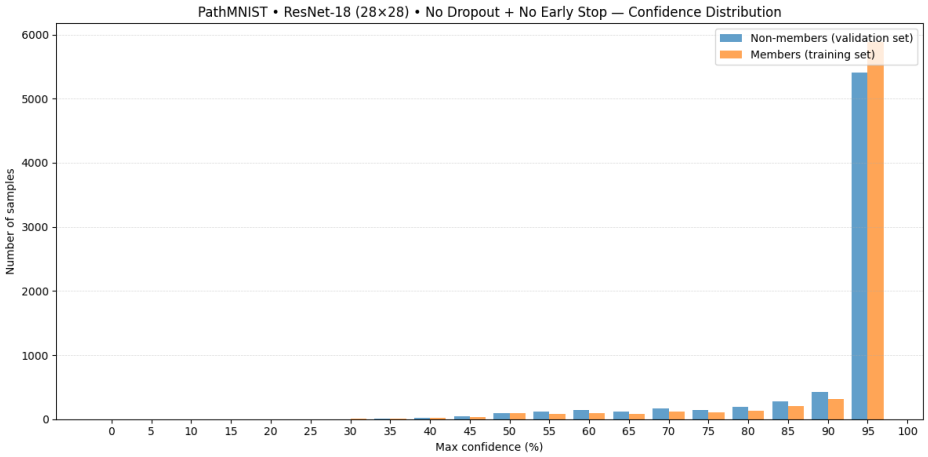


Figure A.195: Confidence distributions for members vs non-members (PathMNIST No Regularisation with ε = 1.0).

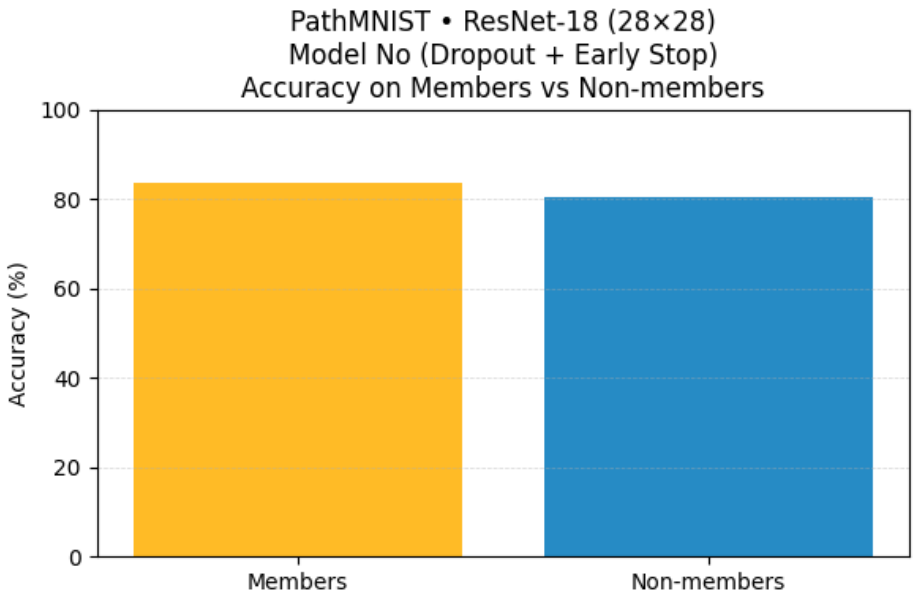
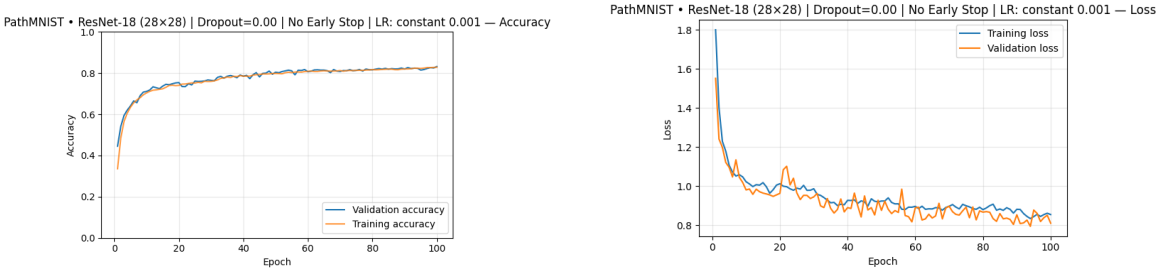


Figure A.196: Accuracy comparison for members and non-members (PathMNIST No Regularisation with ε = 1.0).

ε = 1.2



(a) Accuracy. **(b)** Loss.

Figure A.197: Training and validation performance for the PathMNIST No Regularisation with ε = 1.2 .

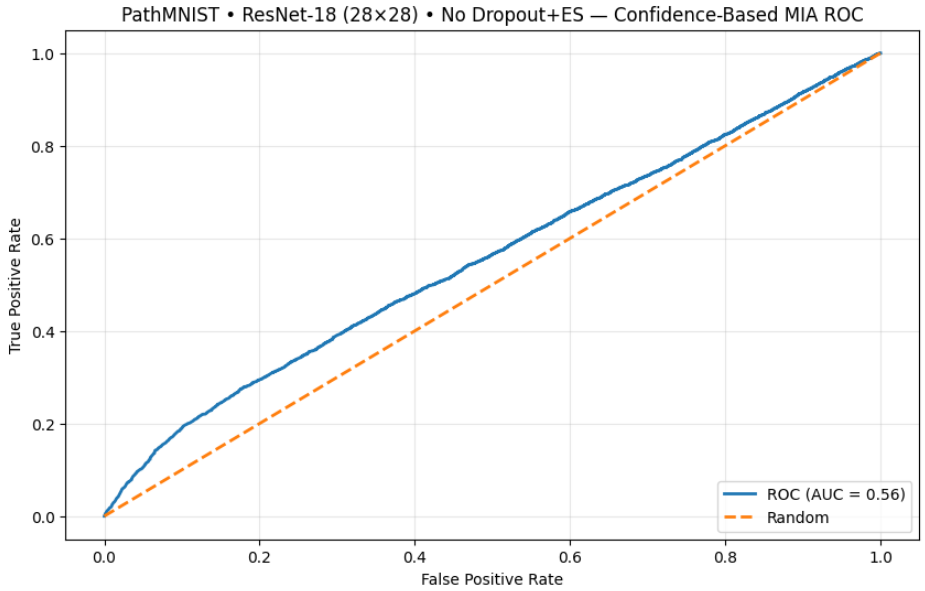


Figure A.198: ROC curve for confidence-based MIA (PathMNIST No Regularisation with ε = 1.2).

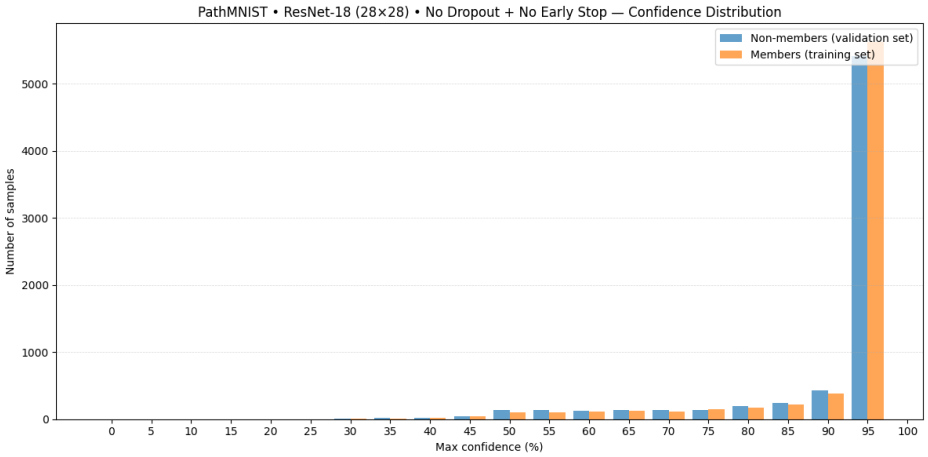


Figure A.199: Confidence distributions for members vs non-members (PathMNIST No Regularisation with ε = 1.2).

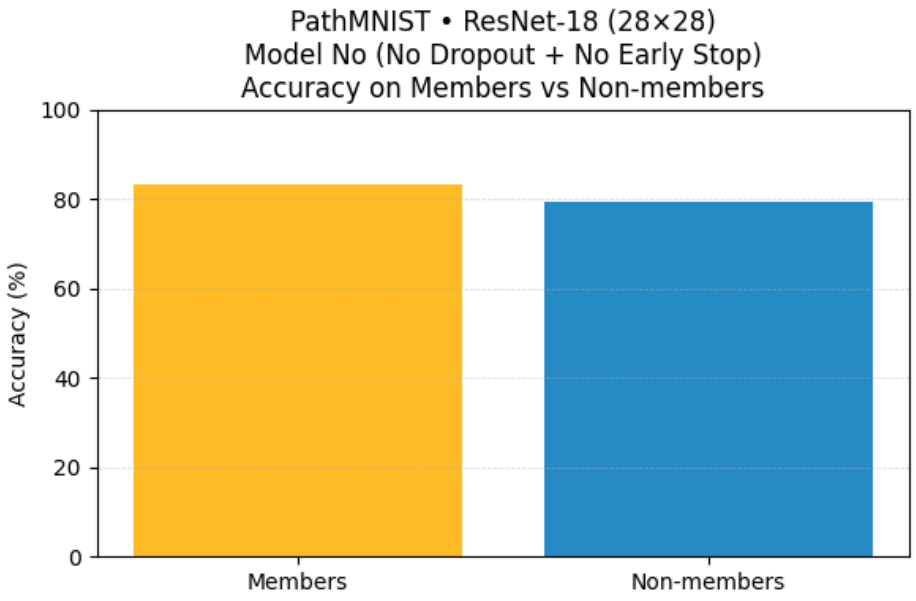
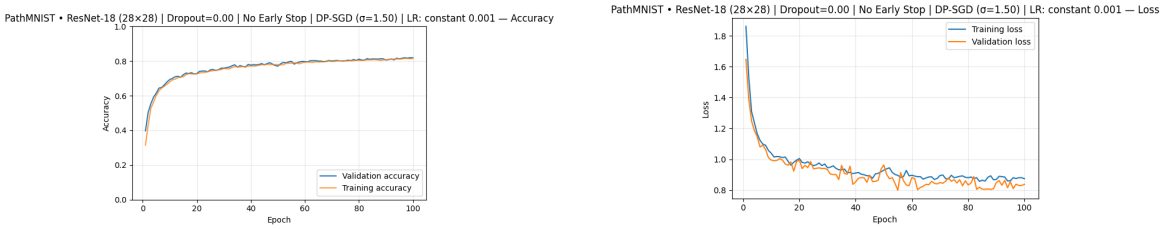


Figure A.200: Accuracy comparison for members and non-members (PathMNIST No Regularisation with ε = 1.2).

ε = 1.5



(a) Accuracy. **(b)** Loss.

Figure A.201: Training and validation performance for the PathMNIST No Regularisation with ε = 1.5 .

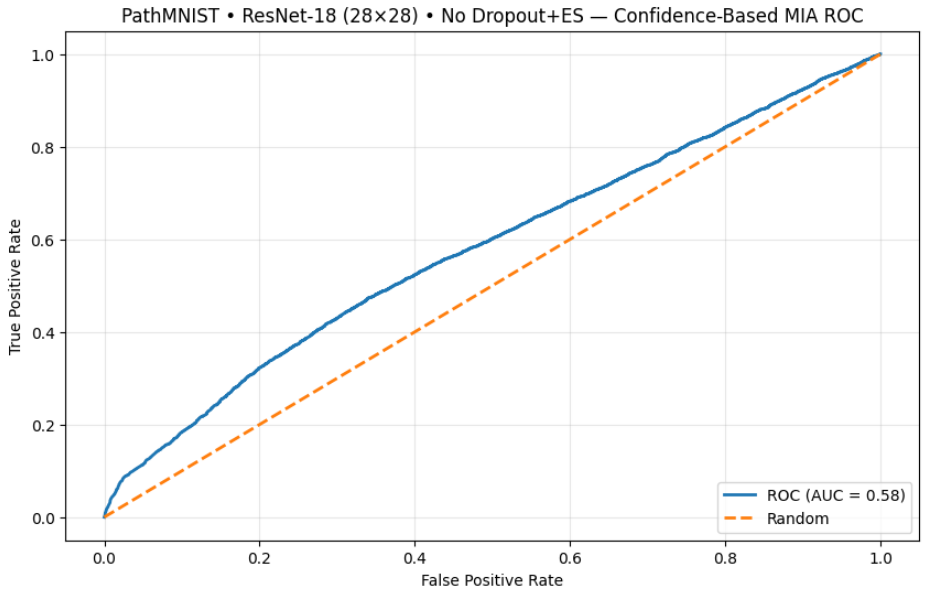


Figure A.202: ROC curve for confidence-based MIA (PathMNIST No Regularisation with ε = 1.5).

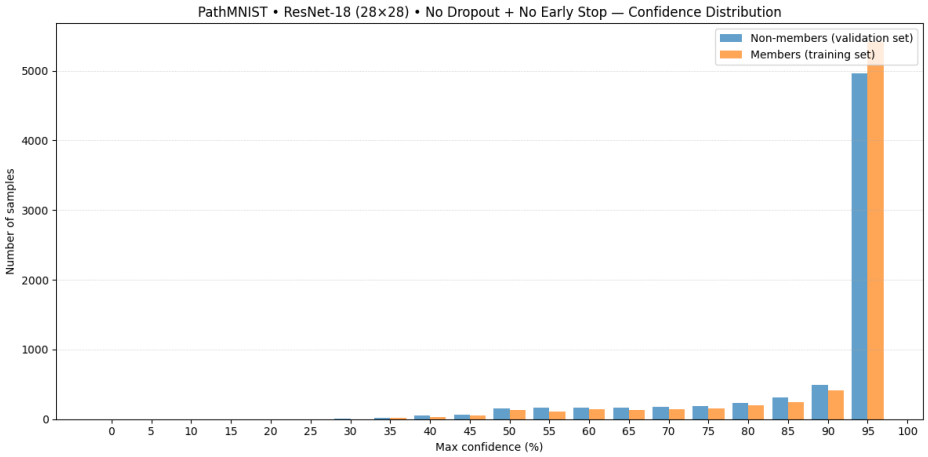


Figure A.203: Confidence distributions for members vs non-members (PathMNIST No Regularisation with ε = 1.5).

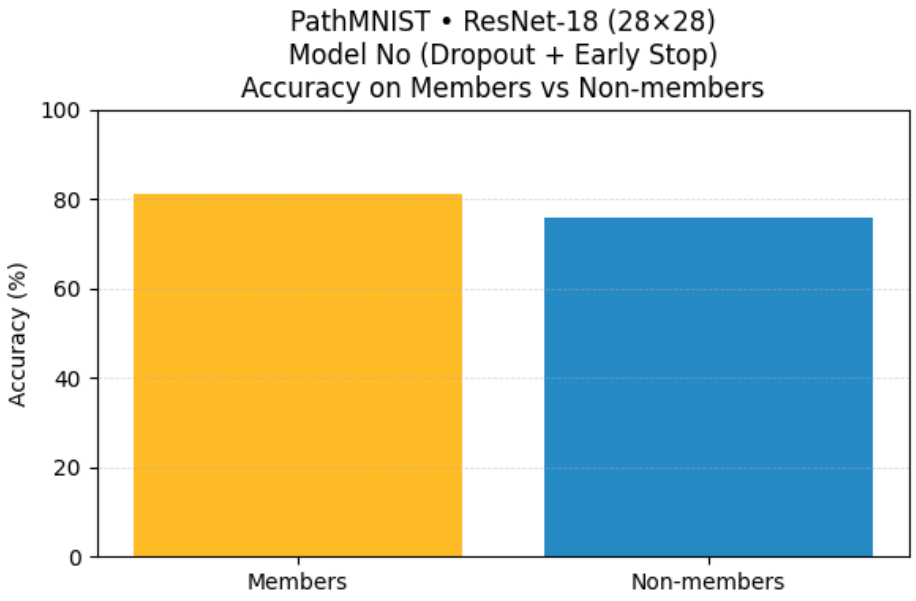
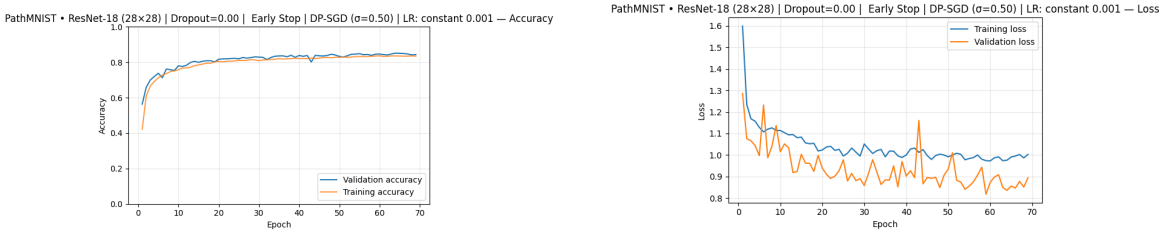


Figure A.204: Accuracy comparison for members and non-members (PathMNIST No Regularisation with ε = 1.5).

A.5.4 DP + Regularisastion

ε = 0.5



(a) Accuracy. **(b)** Loss.

Figure A.205: Training and validation performance for the PathMNIST + Regularisation with ε = 0.5 .

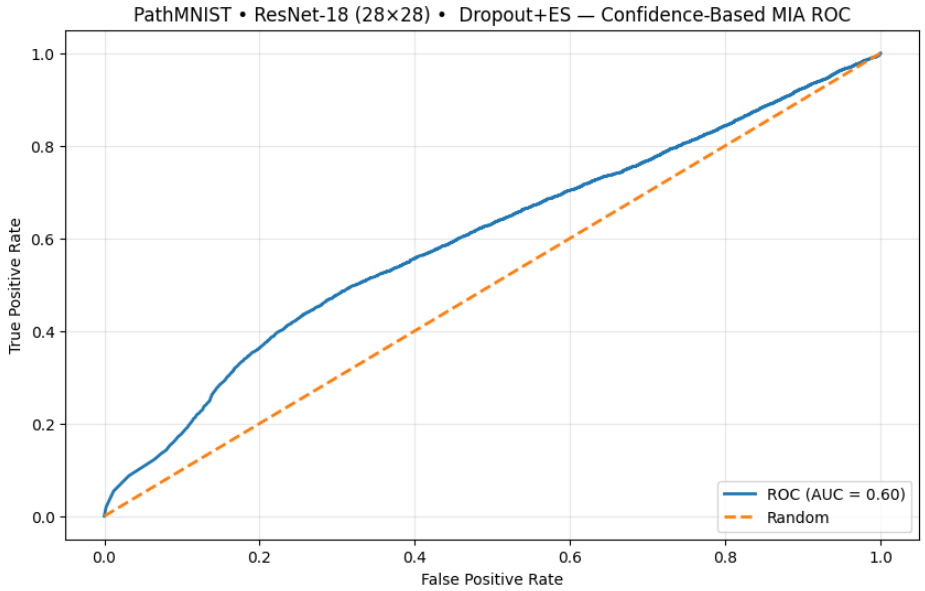


Figure A.206: ROC curve for confidence-based MIA (PathMNIST + Regularisation with ε = 0.5).

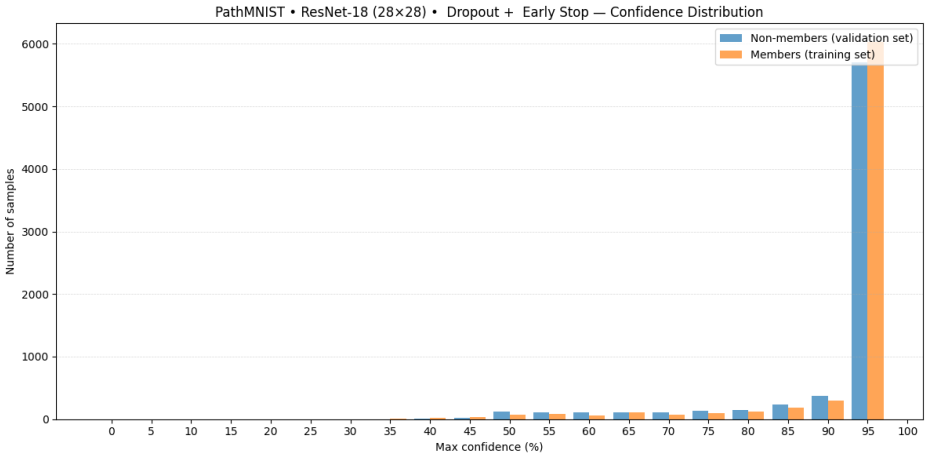


Figure A.207: Confidence distributions for members vs non-members (PathMNIST + Regularisation with ε = 0.5).

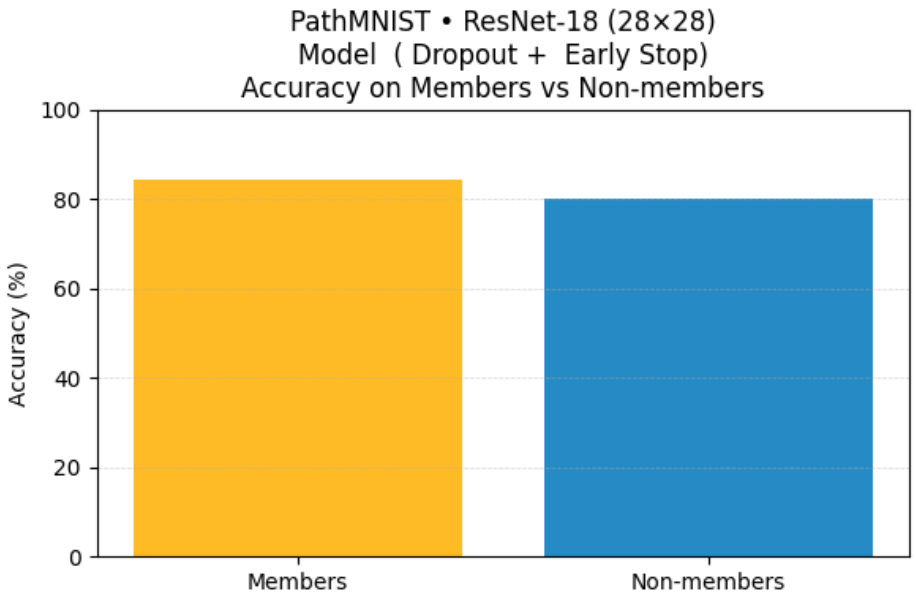
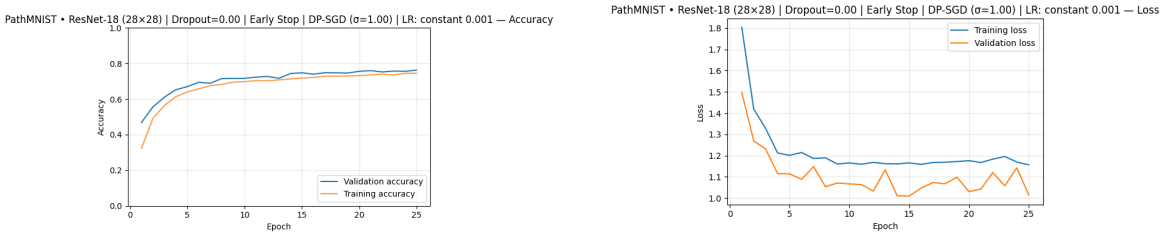


Figure A.208: Accuracy comparison for members and non-members (PathMNIST + Regularisation with ε = 0.5).

ε = 1.0



(a) Accuracy. **(b)** Loss.

Figure A.209: Training and validation performance for the PathMNIST + Regularisation with ε = 1.0 .

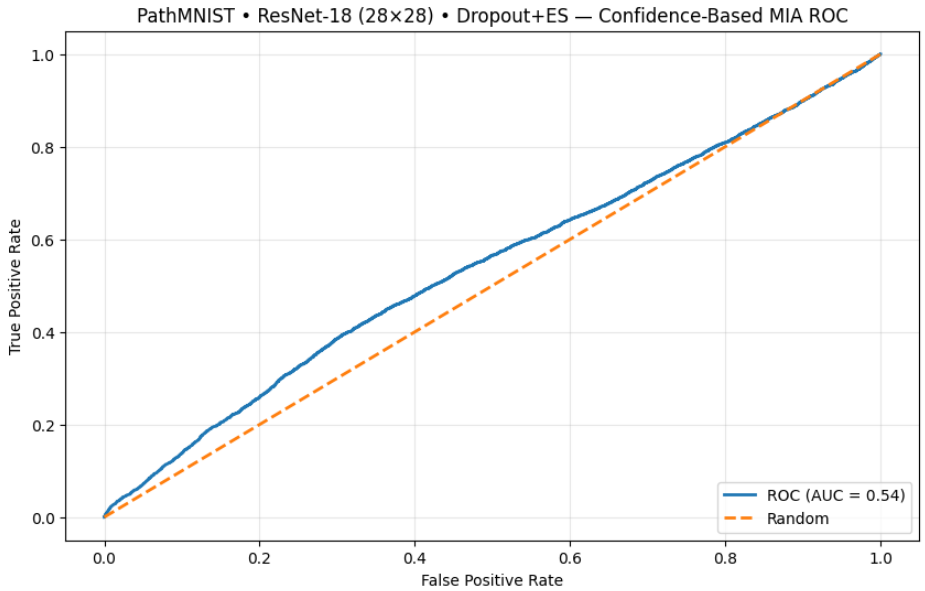


Figure A.210: ROC curve for confidence-based MIA (PathMNIST + Regularisation with ε = 1.0).

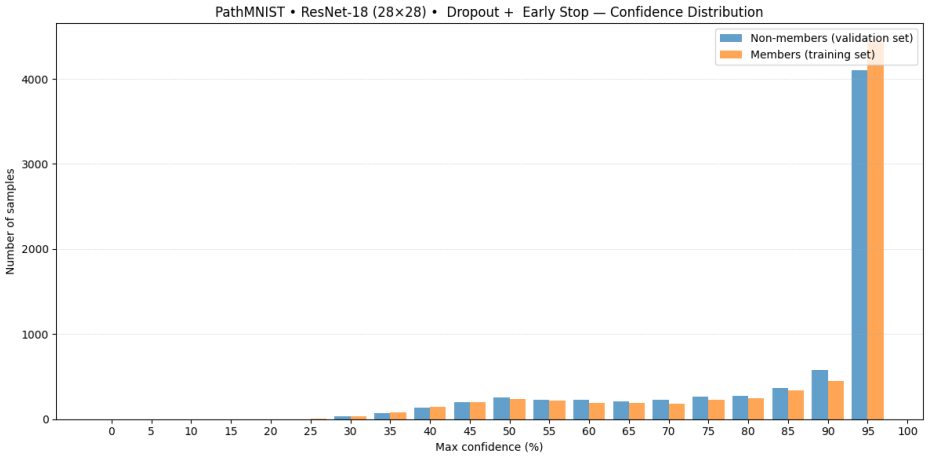


Figure A.211: Confidence distributions for members vs non-members (PathMNIST + Regularisation with ε = 1.0).

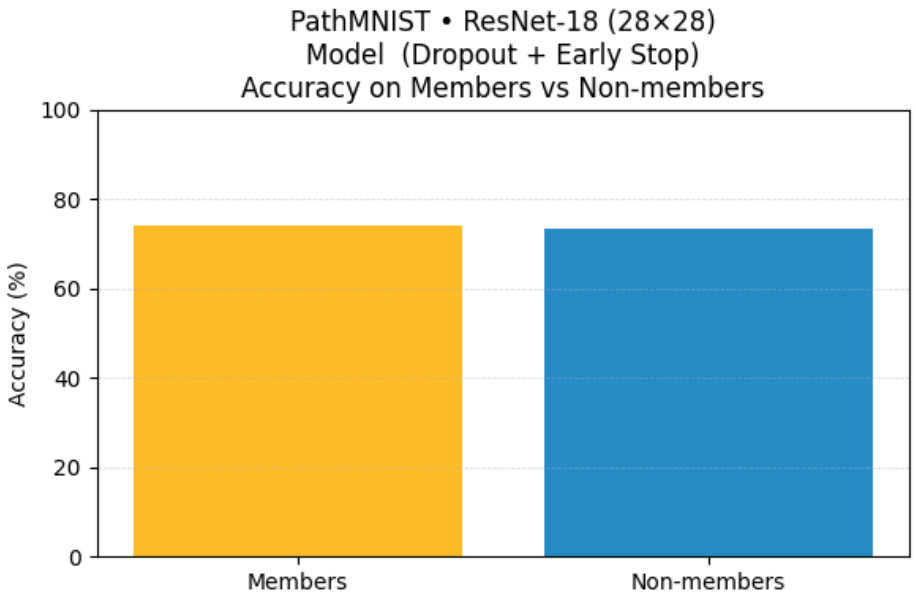
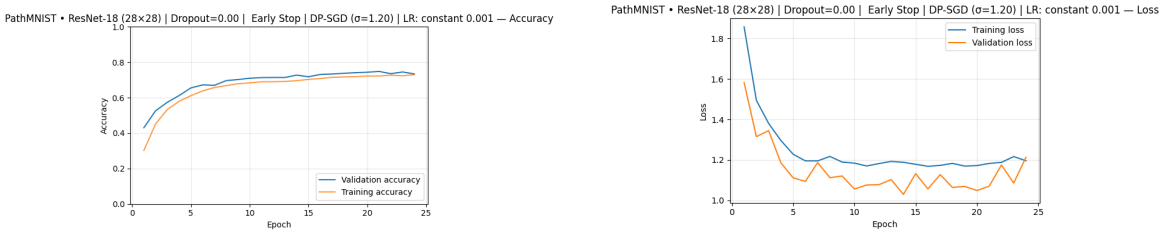


Figure A.212: Accuracy comparison for members and non-members (PathMNIST + Regularisation with ε = 1.0).

ε = 1.2



(a) Accuracy. **(b)** Loss.

Figure A.213: Training and validation performance for the PathMNIST + Regularisation with ε = 1.2 .

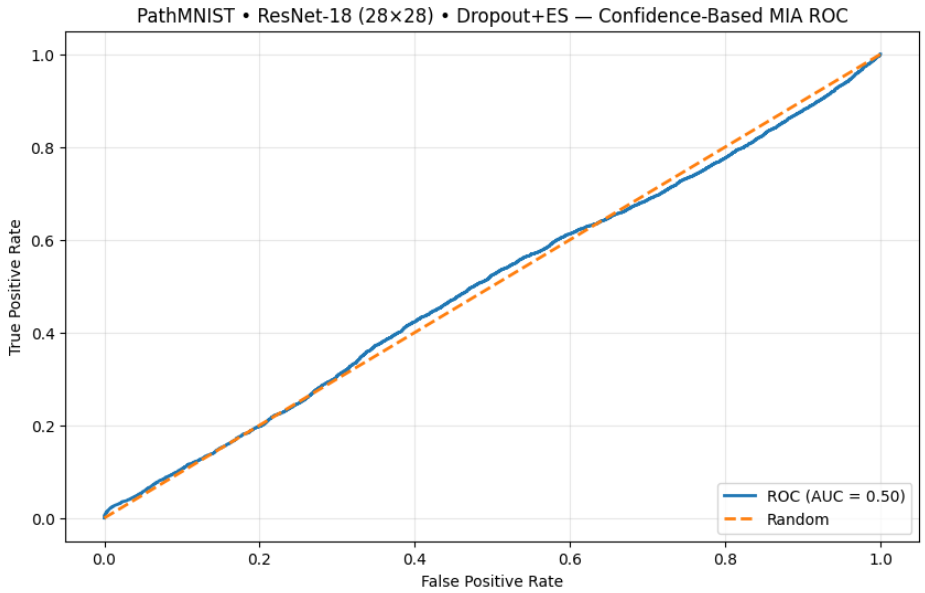


Figure A.214: ROC curve for confidence-based MIA (PathMNIST + Regularisation with ε = 1.2).

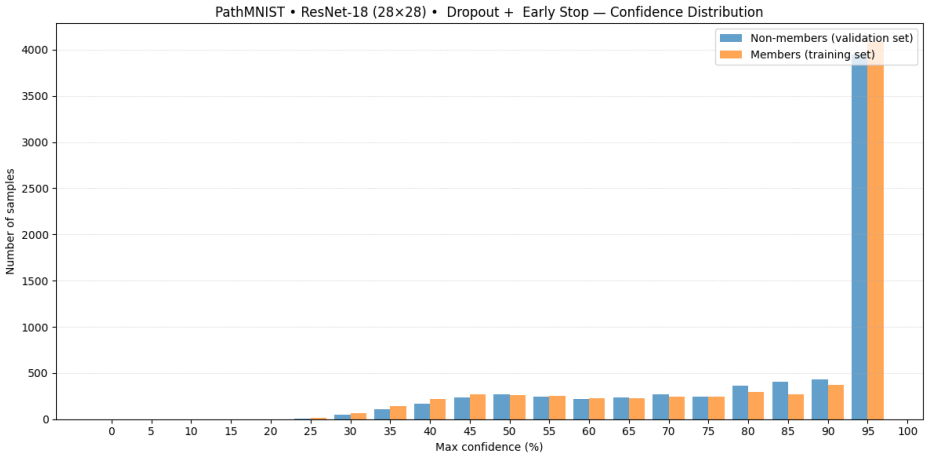


Figure A.215: Confidence distributions for members vs non-members (PathMNIST + Regularisation with ε = 1.2).

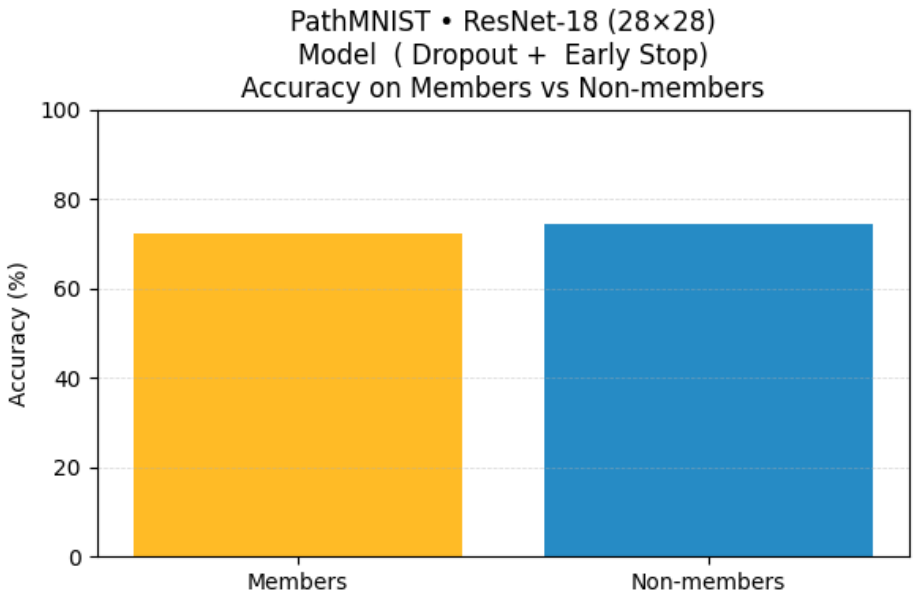
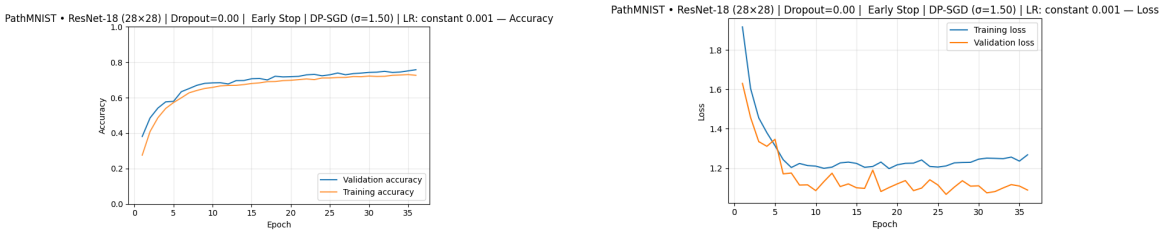


Figure A.216: Accuracy comparison for members and non-members (PathMNIST + Regularisation with ε = 1.2).

ε = 1.5



(a) Accuracy. **(b)** Loss.

Figure A.217: Training and validation performance for the PathMNIST + Regularisation with ε = 1.5 .

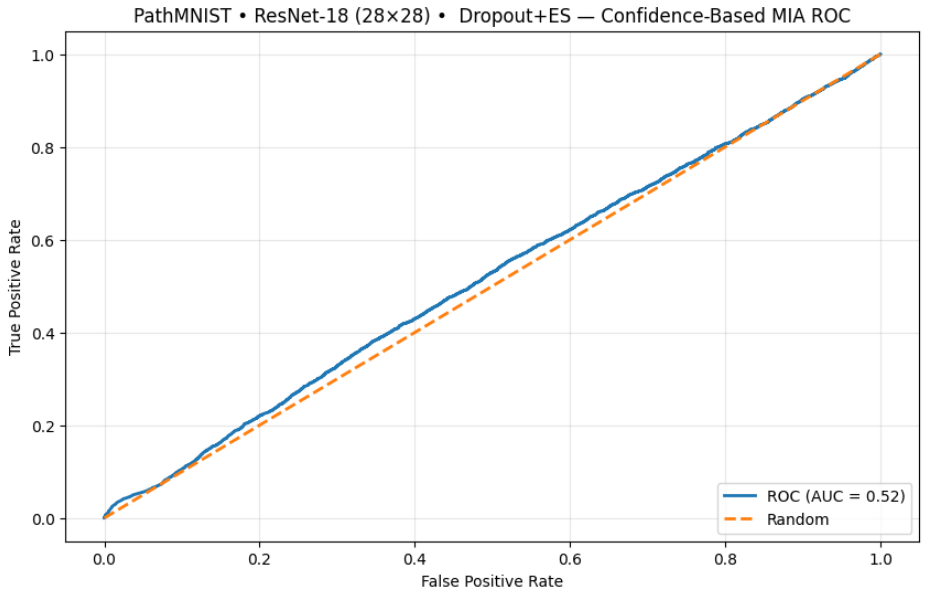


Figure A.218: ROC curve for confidence-based MIA (PathMNIST + Regularisation with ε = 1.5).

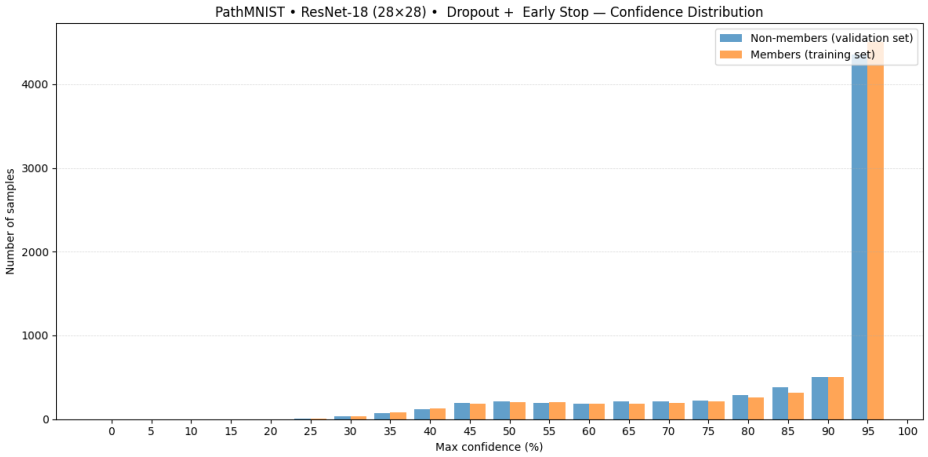


Figure A.219: Confidence distributions for members vs non-members (PathMNIST + Regularisation with ε = 1.5).

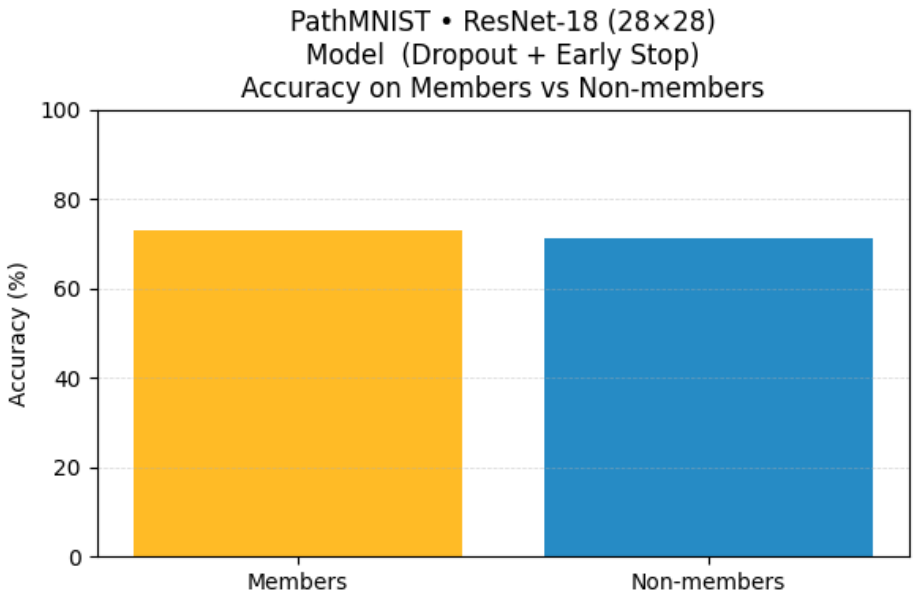


Figure A.220: Accuracy comparison for members and non-members (PathMNIST + Regularisation with ε = 1.5).

Appendix B

Shadow + Transfer Notebooks

B.1 Shadow Experiments

B.1.1 CIFAR-10

Table B.1: Google Colab notebooks for Shadow attack experiments on CIFAR-10.

Configuration	Notebook Link
Baseline	https://colab.research.google.com/drive/1ShAt6KxBR2LvN9hEwWDajfgME0G5NLNd?usp=sharing
Regularised	https://colab.research.google.com/drive/1mkygpINc0Sr4-RIcJTqxVLqyEvTy_6bU?usp=sharing
DP08–NoReg	https://colab.research.google.com/drive/1JdC9smDdlgJN4_mfKgNm-cTxWo4Qw9Dm?usp=sharing
DP12–NoReg	https://colab.research.google.com/drive/1JdC9smDdlgJN4_mfKgNm-cTxWo4Qw9Dm?usp=sharing
DP08–Reg	https://colab.research.google.com/drive/1OB2bEDxzb-c2DJAwy9mqjSLoYf9m5ICs?usp=sharing
DP12–Reg	https://colab.research.google.com/drive/1PuvyNk7MopcKCHaIPD-FswJ9D2FntCC0?usp=sharing

B.1.2 PathMNIST

Table B.2: Google Colab notebooks for Shadow attack experiments on PathMNIST.

Configuration	Notebook Link
Baseline	https://colab.research.google.com/drive/19u6QZbx6tDefgVS5Jn4OWSbD4hdqsIGI?usp=sharing
Regularised	https://colab.research.google.com/drive/1yD6weE0c5s091i-mMtT-yDmPRPYPYm2g?usp=sharing
DP08–NoReg	https://colab.research.google.com/drive/1XU4V48lBQL-negEmmWLBP9-l9NDWhDww?usp=sharing
DP12–NoReg	https://colab.research.google.com/drive/1f46EpEGbwthLfhOw68vcDcSHRnKTO58p?usp=sharing
DP08–Reg	https://colab.research.google.com/drive/1Gg4k0qtXSMn1YTl9j_MMJPfXKuNgMSp6?usp=sharing
DP12–Reg	https://colab.research.google.com/drive/1VnvDArKagV86ZCIh7UsdX0ZC3y-KT-JY?usp=sharing

B.2 Transfer Experiments

B.2.1 CIFAR-10

Table B.3: Google Colab notebooks for Transfer attack experiments on CIFAR-10.

Configuration	Notebook Link
Baseline	https://colab.research.google.com/drive/1XIrszU1qkueF227rMIjHxl9y4iAU-kMF?usp=sharing
Regularised	https://colab.research.google.com/drive/14sGxCbCuGaFF0yvsaUAgL3fO_TKpdt2T?usp=sharing
DP08–NoReg	https://colab.research.google.com/drive/1yd-BOwXR2X0Wqd_BCsUc1yXZoMHtCD_k?usp=sharing
DP12–NoReg	https://colab.research.google.com/drive/1EDv8eHtIRWuou1av8Zt7HhHsjgLQ8_J9?usp=sharing
DP08–Reg	https://colab.research.google.com/drive/1f5D2AmEfr0iGuUMaazKA-33-oQNb46YT?usp=sharing
DP12–Reg	https://colab.research.google.com/drive/19r6RAxqVpul_xP3syFThZJ-cOEF1B_1K?usp=sharing

B.2.2 PathMNIST

Table B.4: Google Colab notebooks for Transfer attack experiments on PathMNIST.

Configuration	Notebook Link
Baseline	https://colab.research.google.com/drive/1SJT_bQTqF6xTn5H-glOjY0Qilxj_lx4j?usp=sharing
Regularised	https://colab.research.google.com/drive/1VP37_CtfcycKgMeGa-k2OZmNZyYlUmLM?usp=sharing
DP08–NoReg	https://colab.research.google.com/drive/10R2W4z0Se_b_3sz0qnmtOIELntIOaDNg?usp=sharing
DP12–NoReg	https://colab.research.google.com/drive/1XgQvbU2hnTDX1os33F2JHoFHBgpDj-bm?usp=sharing
DP08–Reg	https://colab.research.google.com/drive/1lutICzPEg0M4VmmliJVfxvG_hQsRVh6D?usp=sharing
DP12–Reg	https://colab.research.google.com/drive/1zhbJD8AGEvvLdFIHbQCzDD2Qnkxa9QHa?usp=sharing